版权注意事项：
1、书籍版权归作者和出版社所有
2、本PDF仅限用于个人获取知识，进行私底下的知识交流
3、PDF获得者不得在互联网上以任何目的进行传播
4、如觉得书籍内容很赞，请购买正版实体书，支持作者
5、请于下载PDF后24小时内删除本PDF。

2折
购买配套
视频课程

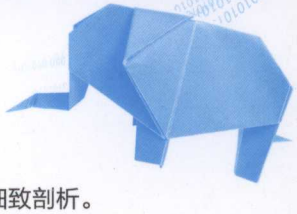
享誉国内的顶尖**网络实战专家**、多届国内**IT图书**最佳原创作者
“网管师”认证教材唯一指定作者**王达老师**倾心编写

新版

Computer Network
Technologies in Depth

深入理解计算机网络

王达 著



- **系统、专业、深入**
  对体系结构中各层主要功能的实现原理和主要通信协议的工作原理都进行了示例化细致剖析。
- **化繁为简，通俗易懂**
  为大量晦涩的技术原理配备了令人耳目一新的形象比喻，讲解生动、示例详细、图表直观，让复杂的原理变得通俗易懂。
- **丰富的教学/自学资源**
  教学PPT和配套视频课程；读者QQ群516844263；微信号windanet；微信公众号windanetclass。

中国水利水电出版社
www.waterpub.com.cn

王达



◎《Cisco/H3C交换机配置与管理完全手册》（第二版）：51CTO“年度最受读者喜爱的原创IT图书”、思科中文技术社区“最值得阅读的十大图书”、至顶网“十大最受欢迎技术图书”。

◎《路由器配置与管理完全手册——H3C篇》：至顶网“最受欢迎计算机图书”。

◎《路由器配置与管理完全手册——H3C篇》和《金牌网管师——网络工程方案规划与设计》：51CTO“年度读者最喜爱的原创IT图书”。

◎《网管员必读——超级网管经验谈》（第二版）：互动出版网“我最喜爱的技术图书”。

◎《网络工程师必读——网络工程基础》：51CTO“读者最喜爱的技术图书”。

◎《网管员必读》系列图书：第13届国际图书博览会“输出版权优秀图书奖”、中国书刊发行业协会等单位联合颁发的“全行业优秀畅销品种奖”、电子工业出版社“最佳品牌奖”、华储网“读者最喜爱的IT图书”称号。

个人所获荣誉

51CTO“最佳原创IT图书作者”称号（连续五届）

电子工业出版社“优秀作者”称号

电子工业出版社易飞思公司“最佳贡献奖”

第二书店“输出荣誉奖”

华储网“最佳IT图书作者奖”

深入理解计算机网络

王达 著

中国水利水电出版社
www.waterpub.com.cn
·北京·

内 容 提 要

本书由国内资深网络技术专家、知名IT图书作家、全国网管技能水平考试认证专家王达老师撰写。本书上一版上市三年多来一再重印（7次印刷），好评不断，被许多高校选作教材，同时被许多读者认为是国内最通俗易懂、最系统的计算机网络原理类图书，甚至被一些读者赞誉为“国内原创、更加通俗的《TCP/IP详解》”。

本版是在上一版的基础上的全面重写，不仅新增了大量内容，而且结合最新技术和笔者获得的新经验对原有内容进行了全面、细致的修改，使本书内容无论在专业性、经验性还是实用性方面都得到了极大地增强。在内容方面，与上一版一样，本书仍以TCP/IP、OSI/RM体系结构为主线，结合当前最新的LAN（如2.5G和5G以太网等）和WLAN技术（如802.11ac和802.11ad等），以类比的方式全面、系统、深入地剖析了计算机网络体系结构中各主要知识点、各层主要功能和通信协议的实现原理；在可读性方面，书中处处体现了作者扎实的专业功底和丰富的专业经验，通过许许多多生动的比喻、568幅图表和大量的讲解示例使许多原本复杂的技术原理在作者的笔下变得简单，化枯燥于无形，同时极大地提高了学习效率。在计算机网络原理方面，本书可使你真正“一本在手，别无所求”。

本书既适合想全面、系统地学习计算机网络技术的网络管理员和网络工程师们，又适合高等院校的老师和学生们用作系统学习计算机网络技术的教材。为了让各位更好地掌握本书所讲的内容，本书为高校老师免费提供配套的PPT，所有读者均可优惠购买配套的视频课程（60多小时），另外提供了492道自测题并在书后附有参考答案，方便读者自己检验学习效果。

图书在版编目（CIP）数据

深入理解计算机网络 / 王达著. -- 北京 : 中国水利水电出版社, 2017.1
ISBN 978-7-5170-4506-9

Ⅰ. ①深… Ⅱ. ①王… Ⅲ. ①计算机网络 Ⅳ. ①TP393

中国版本图书馆CIP数据核字(2016)第149290号

策划编辑：周春元　　责任编辑：张玉玲　　封面设计：李　佳

书　　名	深入理解计算机网络　SHENRU LIJIE JISUANJI WANGLUO
作　　者	王达　著
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路1号D座　100038） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）68367658（营销中心）、82562819（万水）
经　　售	全国各地新华书店和相关出版物销售网点
排　　版	北京万水电子信息有限公司
印　　刷	三河市铭浩彩色印装有限公司
规　　格	184mm×240mm　16开本　42.5印张　990千字
版　　次	2017年1月第1版　2017年1月第1次印刷
印　　数	0001—4000册
定　　价	98.00元

凡购买我社图书，如有缺页、倒页、脱页的，本社营销中心负责调换

版权所有 · 侵权必究

I

前　言

本书的写作动机

本书上一版上市三年多来一再重印（7次印刷），好评不断，被许多高校选作教材，同时被许多读者认为是国内最通俗易懂、最系统的计算机网络原理类图书，甚至被一些读者赞誉为“国内原创、更加通俗的《TCP/IP详解》”。在此借这个机会向广大读者朋友表示我最真诚的感谢。

三年多来，不仅计算机网络技术在不断革新，笔者这几年在新书创作和视频课程录制的过程中也积累了许多新的宝贵经验。另外，在这几年中，也有许多读者朋友对我提出了宝贵的建议，特别是一些高校的老师，希望我的这本书能更贴近高校教学实际。为了能使这本书更专业、更实用、更贴近高校教学实际，帮助更多读者和高校学生深入地学习计算机网络技术，笔者决定对本书上一版进行改版。

本次改版绝不是简单的修订，而是真正的全面改写，在删除上一版中与计算机网络原理关系不是特别密切的各种交换机和路由器技术原理内容的基础上，新增了大量非常重要的内容，如浮点数的运算、802.11ac、802.11ad、FTP和SNMP协议工作原理等。同时对原有内容进行了全面、细致的修订，新增大量贴切、生动的比喻，使其描述更专业、更通俗易懂、更贴近实际的网络管理和网络工程工作。

本书的主要特色

本书虽然与市场上其他众多计算机网络原理类图书在主题和总体架构上是类似的，都是以计算机网络体系结构为主线而展开，但是本书却与同类图书有着非常大的不同，具有非常鲜明的特色，这也是本书上一版上市后就立即得到众多读者的大力支持和高度肯定，得到许多高校师生的欢迎，并且一直保持领先的根本原因。本书的主要特色体现在以下几个方面：

（1）系统、专业、深入。

这绝对不是口号，而是几乎所有看过本书上一版的朋友的共识。有许多读者购买本书之前通过查看目录感觉主要的知识点与其他同类图书类似，就是不明白为什么我的这本书的篇幅却厚如此之多，几乎是它们的两倍。但在他们真正看了书中的内容后终于明白，无论在内容的系统性、通俗性，还是在专业性和深入性方面，本书远非其他同类书可比。也正因为如此，本书

上一版被许许多多读者给予了极高评价，这点大家可以在当当网、京东网等网店中查看。

说实话，这类图书其实是最难写的，没有扎实的专业功底是很难写好的。现在市场上绝大多数同类教材，要么似是什么都写到了，但基本上都停留在概念描述的层次，相当于网上“百科”，读者不可能从中得到深入的学习；要么只写了一些作者掌握的少数主要内容，读者根本无法从中进行系统的学习。

在内容系统性方面，本书几乎包括了计算机网络体系结构中所有方面的主要知识点，从最基本的数制、计算机网络基础、计算机网络体系结构的发展历程、数据通信原理，再到从最低的物理层到最高的应用层各项功能原理、主要通信协议工作原理等无所不包，最后还介绍了当前最新的技术，如 2.5G 和 5G 以太网、802.11ac 和 802.11ad WLAN 技术等；在深入性方面，本书对体系结构中各层主要功能的实现原理和主要通信协议的工作原理都进行细致的示例化剖析，这是从其他同类图书中极难获得的。

至于专业性方面那就更是困难了。有许多读者经常问我一些似是而非的问题，那都是他们从其他图书或者课程中看到的，专业人士一看就知道是错误的。其实这类现象笔者也在一些老师的课程中看到过许多，究其原因就是这些作者或老师对整个计算机网络通信原理或者一些技术原理的理解不正确、不深入。笔者这二十多年来专注的只有一个方向，那就是计算机网络。而且在这二十多年来不断学习，也出版了许多这方面的图书，对整个计算机网络通信原理有着深入的理解，同时也积累了大量独家经验，这些都是我最宝贵的财富。

（2）化繁为简，通俗易懂。

对于一本主要介绍技术原理类的图书，如何实现通俗易懂的目的也是非常重要的，因为大多数读者是处于初级水平，甚至是零基础，如果不通俗地进行阐述读者们是无法接受的。但是要对这些复杂的技术原理一一进行通俗化的阐述也是极不容易做到的，必须建立在非常专业的基础之上，否则因为比喻不合适就可能造成读者的错误理解。

笔者写本书上一版时花了整整一年的专职时间，本版改写又花了几个月时间，尽管我以前也写过许多本同类图书。其中相当大一部分时间就是用在为各种技术原理思考合适的比喻、讲解示例和图表描述上，其目的当然是想让复杂的原理变得通俗易懂。本版图书有近百个以现实生活事件为例的比喻、568 幅图表、30 多个讲解示例，这应该都是国内同类图书所无法比拟的。

（3）丰富的教学/自学资源。

为了更好地帮助高校老师教学，笔者为本书编制了新版教学 PPT（比上一版的内容更全面、更实用），凭本书学生教材采购单复印件就可免费获取。另外，为了帮助广大读者朋友自学直接在各章后面附上了新编写的大量自测题（共 492 道）并在书后附有参考答案。如果你还觉得学习效率不够高，笔者还为大家准备了本书配套的视频课程（60 多小时），可凭本书携带的“学习码”以 2 折（仅 150 元，原价 750 元）购买下载版，加入后面为大家提供的读者 QQ 群、本人微信 windanet 咨询。

本书适用的读者对象

本书的内容非常系统、通俗、专业和深入，适合各层次的读者，具体如下：

- 各高等院校的计算机网络专业学生
- 想从事各种计算机网络运维工作的朋友
- 还没有系统、深入学习计算机网络原理的朋友
- 看不懂一些同类图书的朋友
- 对原来购买的同类图书或者学校教材失望的朋友们
- 工作了好几年，技能水平却一直无法提高的朋友
- 想让自己的职业和技能水平上一个新台阶的朋友
- 整天抱怨工资太低，却不清楚自己当前技能水平的朋友

本书的服务资源

本书由王达主笔并统稿，参加编写、校验和排版的人员有：何艳辉、王珂、沈芝兰、马平、何江林、刘凤竹、卢京华、周志雄、洪武、高平复、周建辉、孔平、尚宝宏、姚学军、张磊、刘学、李翔、王娇、李敏、吴鹏飞、宋希岭、刘中洲、潘朝阳、刘伟、曾平辉、李京杨、张跃、周平辉、王新宇、王薄、韩大为、宋宝强、史鹏宇、陆伟等。

中国水利水电出版社的各位领导和编辑老师为本书出版付出了非常艰辛的劳动，在此一并表示由衷的谢意。

由于编者水平有限，尽管我们花了大量时间和精力校验，书中仍可能存在一些错误和瑕疵，敬请各位批评指正，万分感谢！

另外，本书读者可以通过以下渠道获取更多的学习资源并享受相关服务：

（1）多个专家博客和认证微博。

51CTO 专家博客：http://winda.blog.51cto.com

CSDN 专家博客：http://blog.csdn.net/lycb_gz

新浪认证微博：weibo.com/winda

腾讯认证微博：t.qq.com/winda2010

（2）视频课程中心。

51CTO 学院课程中心：http://edu.51cto.com/lecturer/user_id-55153.html

CSDN 学院课程中心：http://edu.csdn.net/lecturer/74

备注：大家需要我的视频课程时可直接在以上课程中心购买，也可直接向我购买下载版，还可购买下载版视频课程终身会员，获得我录制的所有视频课程。另外，我每年还将举办两期左右的华为、Cisco 或 H3C 的各类认证网上直播视频培训，可加入以下读者 QQ 群或学员 QQ 群向我了解：

- 读者和学员 QQ 群：为了能为大家提供更好的服务，同时也方便大家相互交流、学习，笔者为本书读者专门创建了一个新的 QQ 读者群：516844263。对于购买了我的视频课程的学员朋友，可加入以下两个视频课程学员 QQ 群之一：398772643 和 241903278。
- 个人微信和公众号：本人微信为 windanet，读者微信公众号为 windanetclass。凭本书学习码加入本人微信后可以 2 折（折后价 150 元，原价 750 元）购买本书配套视频课程，共计 60 多小时。还可在笔者的淘宝网店中优惠购买我的其他课程：https://shop33132415.taobao.com/。

II 目录

前言

第 1 章　数制基础和机器数运算 …… 1

1.1　数制概述 …… 2
1.1.1　常见数制类型及表示方法 …… 2
1.1.2　不同数制之间的对应关系 …… 4
1.2　不同数制间的相互转换 …… 5
1.2.1　非十进制数转换成十进制数 …… 5
1.2.2　十进制数转换成非十进制数 …… 6
1.2.3　非十进制数之间的相互转换 …… 10
1.3　机器数基础 …… 11
1.3.1　机器数的真值 …… 12
1.3.2　机器数的字长 …… 12
1.4　机器数的编码形式 …… 13
1.4.1　原码 …… 13
1.4.2　补码 …… 14
1.4.3　反码 …… 15
1.4.4　阶码 …… 16
1.4.5　移码 …… 16
1.5　机器数的分类 …… 17
1.5.1　定点数 …… 18
1.5.2　浮点数 …… 19
1.5.3　IEEE 754 浮点数的分类 …… 20
1.5.4　IEEE 754 浮点数的计算 …… 21
1.5.5　浮点数表示形式的转换 …… 22
1.6　二进制数的运算 …… 23
1.6.1　二进制数的算术运算 …… 23
1.6.2　补码的加/减法运算 …… 26
1.6.3　二进制数的逻辑运算 …… 28
1.7　浮点数的加/减法运算 …… 30
1.7.1　对阶 …… 30
1.7.2　尾数运算 …… 31
1.7.3　规格化处理 …… 31
1.7.4　舍入处理 …… 32
1.7.5　溢出处理 …… 33
1.8　信息编码 …… 33
1.8.1　西文编码 …… 34
1.8.2　中文编码 …… 35
1.9　课后自测题 …… 36
1.9.1　填空题 …… 36
1.9.2　选择题（可多选） …… 37
1.9.3　计算题 …… 38

第 2 章　计算机网络概述 …… 40

2.1　计算机网络综述 …… 40
2.1.1　计算机网络的发展历史 …… 41
2.1.2　计算机网络的基本组成 …… 47
2.1.3　计算机网络的主要应用 …… 49
2.2　计算机网络的分类 …… 51

2.2.1 按网络所覆盖的地理范围分 …… 51
2.2.2 按网络管理模式分 …… 53
2.2.3 按传输方式分 …… 57
2.3 计算机网络拓扑结构 …… 59
2.3.1 网络拓扑结构相关基本概念 …… 59
2.3.2 星型拓扑结构 …… 60
2.3.3 环型拓扑结构 …… 64
2.3.4 总线型拓扑结构 …… 69
2.3.5 树型拓扑结构 …… 73
2.3.6 混合型拓扑结构 …… 74
2.3.7 网状拓扑结构 …… 76
2.3.8 无线局域网的两种拓扑结构 …… 78
2.4 课后自测题 …… 79
2.4.1 填空题 …… 79
2.4.2 选择题（可多选） …… 80
第 3 章 计算机网络体系结构 …… 83
3.1 计算机网络体系结构发展历程 …… 83
3.2 OSI/RM 体系结构 …… 87
3.2.1 OSI/RM 的七层结构 …… 87
3.2.2 OSI/RM 的层次划分原则 …… 89
3.3 TCP/IP 协议体系结构 …… 89
3.3.1 TCP/IP 体系结构层次 …… 90
3.3.2 OSI/RM 和 TCP/IP 体系结构的比较 …… 91
3.3.3 局域网体系结构 …… 93
3.4 网络体系结构的设计考虑 …… 94
3.4.1 体系结构的设计考虑 …… 94
3.4.2 体系结构的层次划分考虑 …… 95
3.4.3 主机系统层 …… 96
3.4.4 体系结构分层的好处 …… 97
3.5 例说体系结构各层主要用途 …… 98
3.5.1 各层角色的类比 …… 98
3.5.2 各层主要作用的类比 …… 99
3.6 计算机网络通信原理 …… 102
3.6.1 计算机网络通信流程 …… 102
3.6.2 对等会话原理 …… 103
3.6.3 各层的数据传输单元 …… 104
3.6.4 协议封装和解封装 …… 105
3.7 网络体系结构中的通信协议 …… 107
3.7.1 理解计算机网络通信协议 …… 107
3.7.2 网络通信协议的三要素 …… 108
3.8 课后自测题 …… 109
3.8.1 填空题 …… 109
3.8.2 选择题（可多选） …… 110
第 4 章 计算机网络通信基础 …… 113
4.1 数据通信系统基本模型 …… 113
4.2 信号传输系统分类 …… 114
4.2.1 基带与频带 …… 115
4.2.2 数字基带传输系统和数字频带传输系统 …… 115
4.2.3 宽带传输和宽带传输系统 …… 118
4.3 数据传输分类 …… 118
4.3.1 数据传输方式 …… 118
4.3.2 数据传输模式 …… 120
4.3.3 数据通信模式 …… 122
4.4 数据传输速率与信道带宽 …… 125
4.4.1 传输速率与信道带宽的基本概念 …… 125
4.4.2 数字信号不失真传输的最大传输速率限制 …… 126
4.4.3 模拟信号不失真还原的最小采样频率限制 …… 128
4.5 矩形脉冲数字信号基本波形 …… 129
4.5.1 单极性波形和双极性波形 …… 129
4.5.2 归零码和非归零码 …… 130
4.5.3 单极性归零码和双极性归零码 …… 131
4.6 数字基带信号传输码 …… 133
4.6.1 AMI 码的编码原理 …… 134
4.6.2 CMI 码的编码原理 …… 135

4.6.3 HDB3 码的编码原理 …… 136
4.6.4 曼彻斯特码的编码原理 …… 137
4.6.5 差分曼彻斯特码的编码原理 …… 138
4.7 信号调制与解调 …… 140
4.7.1 调制与解调的关键术语 …… 140
4.7.2 2ASK 调制原理 …… 142
4.7.3 2ASK 解调原理 …… 143
4.7.4 2FSK 调制原理 …… 145
4.7.5 2FSK 解调原理 …… 147
4.7.6 2PSK 调制与解调概述 …… 150
4.7.7 2APSK 调制原理 …… 150
4.7.8 2APSK 解调原理 …… 152
4.7.9 2DPSK 调制原理 …… 153
4.7.10 2DPSK 解调原理 …… 155
4.8 课后自测题 …… 157
4.8.1 填空题 …… 157
4.8.2 选择题（可多选） …… 157
4.8.3 综合分析题 …… 161
第 5 章 物理层 …… 162
5.1 物理层基础 …… 162
5.1.1 物理层主要功能 …… 163
5.1.2 机械特性 …… 164
5.1.3 电气特性 …… 166
5.1.4 功能特性 …… 169
5.1.5 规程特性 …… 170
5.1.6 传输介质分类 …… 170
5.2 双绞线 …… 171
5.2.1 双绞线的分类 …… 171
5.2.2 双绞线连接器 …… 173
5.3 光缆 …… 175
5.3.1 光缆的组成 …… 175
5.3.2 光纤的分类 …… 176
5.3.3 光纤模块 …… 178
5.3.4 光纤连接器 …… 180
5.3.5 尾纤与跳纤 …… 182
5.4 串行电缆及连接器 …… 184
5.4.1 串行接口标准概述 …… 184
5.4.2 RS-232 串行接口 …… 185
5.4.3 其他 EIA 标准接口 …… 189
5.4.4 X.21、X.24、X.36 和 EIA-530 接口规范 …… 191
5.5 信道多路复用技术 …… 195
5.5.1 频分复用及其原理 …… 195
5.5.2 时分复用及其原理 …… 198
5.5.3 波分复用及其原理 …… 199
5.6 主要 WLAN 物理层规范 …… 201
5.6.1 WLAN 物理层规范概述 …… 201
5.6.2 IEEE 802.11b 规范主要特性 …… 202
5.6.3 IEEE 802.11a 规范主要特性 …… 204
5.6.4 IEEE 802.11g 规范主要特性 …… 207
5.6.5 IEEE 802.11n 规范主要特性 …… 208
5.6.6 IEEE 802.11ac 规范主要特性 …… 209
5.6.7 IEEE 802.11ad 规范主要特性 …… 210
5.7 课后自测题 …… 211
5.7.1 填空题 …… 211
5.7.2 选择题（可多选） …… 212
第 6 章 数据链路层 …… 215
6.1 数据链路层基础 …… 216
6.1.1 划分“数据链路层”的必要性 …… 216
6.1.2 数据链路层的结构 …… 218
6.2 数据链路层主要功能及实现原理 …… 221
6.2.1 数据链路层服务 …… 221
6.2.2 LLC 子层服务原语 …… 223
6.2.3 数据链路管理 …… 226
6.2.4 数据帧封装与解封装 …… 226
6.2.5 帧组装与帧同步 …… 227
6.2.6 差错控制 …… 229
6.2.7 流量控制 …… 231

6.3 差错控制方案 …… 232
6.3.1 奇偶校验码检错方案 …… 232
6.3.2 循环冗余校验检错方案 …… 234
6.3.3 反馈检测法 …… 236
6.3.4 空闲重发请求方案 …… 237
6.3.5 连续重发请求的回退 N 帧策略 …… 239
6.3.6 连续重发请求的选择重发策略 …… 241
6.4 海明纠错码 …… 243
6.4.1 计算校验位数 …… 243
6.4.2 确定校验码位置 …… 243
6.4.3 确定校验码 …… 244
6.4.4 实现校验和纠错 …… 246
6.5 流量控制 …… 247
6.5.1 XON/XOFF 流量控制方案 …… 247
6.5.2 滑动窗口机制 …… 248
6.6 面向字符的 BSC 同步传输协议 …… 251
6.6.1 BSC 控制字符和数据块结构 …… 251
6.6.2 BSC 协议数据透明传输原理 …… 253
6.7 面向比特的 SDLC 和 HDLC 同步传输协议 …… 254
6.7.1 HDLC 链路结构和操作方式 …… 254
6.7.2 SDLC/HDLC 帧结构 …… 256
6.7.3 SDLC/HDLC 帧类型及其标识方法 …… 259
6.8 面向字符的 PPP 同步传输协议 …… 261
6.8.1 PPP 简介 …… 261
6.8.2 PPP 帧结构和透明传输原理 …… 262
6.8.3 PPP 链路建立、使用和拆除流程 …… 264
6.8.4 PPP 的 PAP/CHAP 身份认证 …… 265
6.9 数据链路层设备及二层交换原理 …… 268
6.9.1 计算机网卡 …… 268
6.9.2 网桥及广播域、冲突域 …… 271
6.9.3 二层交换机 …… 273
6.9.4 二层交换原理 …… 276
6.10 课后自测题 …… 278
6.10.1 填空题 …… 278
6.10.2 选择题（可多选） …… 279
6.10.3 计算题 …… 283

第 7 章 介质访问控制子层 …… 284

7.1 MAC 子层基础 …… 285
7.1.1 两种信道类型 …… 285
7.1.2 MAC 子层概述 …… 287
7.2 CSMA 介质访问控制原理 …… 288
7.2.1 非-坚持算法 …… 289
7.2.2 1-坚持算法 …… 290
7.2.3 p-坚持算法 …… 290
7.3 CSMA/CD 介质访问控制原理 …… 292
7.3.1 CSMA/CD 原理综述 …… 292
7.3.2 冲突检测原理 …… 293
7.3.3 冲突避让原理 …… 295
7.3.4 CSMA/CD 的不足 …… 297
7.4 局域网标准及以太网帧格式 …… 297
7.4.1 IEEE 802 系列局域网标准 …… 297
7.4.2 以太网帧格式综述 …… 299
7.4.3 LLC 帧头部格式 …… 302
7.4.4 SNAP 头部格式 …… 304
7.4.5 MAC 帧格式 …… 306
7.4.6 VLAN 及 QinQ 帧格式 …… 307
7.5 标准以太网规范及体系结构 …… 309
7.5.1 标准以太网规范 …… 309
7.5.2 标准以太网物理层结构 …… 310
7.6 快速以太网规范及体系结构 …… 312
7.6.1 快速以太网规范 …… 312
7.6.2 快速以太网物理层结构 …… 317
7.7 千兆以太网规范及体系结构 …… 318
7.7.1 千兆以太网规范 …… 318
7.7.2 1000Base-T 以太网技术 …… 321
7.7.3 IEEE 千兆以太网物理层结构 …… 323
7.8 万兆以太网规范及体系结构 …… 324
7.8.1 万兆以太网规范 …… 324

7.8.2 万兆以太网物理层结构 …… 327
7.9 WLAN 网络架构及帧格式 …… 329
7.9.1 WLAN 网络基本概念 …… 329
7.9.2 WLAN 网络体系架构 …… 331
7.9.3 WLAN MAC 帧格式 …… 332
7.9.4 CSMA/CA 协议原理 …… 335
7.10 课后自测题 …… 339
7.10.1 填空题 …… 339
7.10.2 选择题（可多选） …… 340
第 8 章 网络层 …… 344
8.1 网络层概述 …… 345
8.1.1 划分网络层的必要性 …… 345
8.1.2 网络层的主要作用 …… 346
8.2 网络层数据交换及相关技术 …… 348
8.2.1 线路交换原理 …… 348
8.2.2 存储－转发原理 …… 350
8.2.3 报文交换原理 …… 351
8.2.4 分组交换原理 …… 352
8.2.5 虚电路交换和数据报交换的比较 … 356
8.3 IPv4 和 IPv6 协议 …… 357
8.3.1 IP 协议基本功能 …… 357
8.3.2 IPv4 的不足 …… 358
8.3.3 IPv6 的主要优势 …… 359
8.3.4 IPv4 数据报头部格式 …… 361
8.3.5 IPv6 数据报头部格式 …… 366
8.3.6 IPv6 扩展报头 …… 368
8.3.7 IPv4 数据报的封装与解封装 …… 369
8.3.8 IPv4 数据报的分段与重组 …… 372
8.4 ARP 协议 …… 374
8.4.1 ARP 报文格式 …… 374
8.4.2 ARP 映射表 …… 376
8.4.3 ARP 地址解析原理 …… 377
8.4.4 免费 ARP …… 379
8.4.5 代理 ARP …… 380
8.4.6 逆向 ARP …… 382
8.5 ICMP 协议 …… 383
8.5.1 ICMP 报头格式 …… 383
8.5.2 ICMP 消息类型 …… 385
8.5.3 常见的 ICMP 应用 …… 385
8.6 IPv6 协议簇中的其他协议 …… 388
8.7 路由和路由算法 …… 390
8.7.1 路由的分类 …… 391
8.7.2 路由算法基础 …… 394
8.7.3 路由表基础 …… 397
8.7.4 路由协议优先级 …… 398
8.7.5 路由算法的设计目标和设计考虑 … 399
8.7.6 深入理解路由与网关的区别 …… 401
8.8 主要路由算法解析 …… 405
8.8.1 最短路径路由算法 …… 405
8.8.2 扩散路由算法 …… 409
8.8.3 距离矢量路由算法 …… 410
8.8.4 链路状态路由算法 …… 414
8.9 网络拥塞控制方法和原理 …… 418
8.9.1 网络拥塞控制方法 …… 418
8.9.2 死锁及其预防 …… 421
8.10 三层交换机与三层交换 …… 423
8.10.1 三层交换机硬件结构 …… 424
8.10.2 三层交换原理 …… 425
8.10.3 三层交换示例 …… 426
8.11 课后自测题 …… 429
8.11.1 填空题 …… 429
8.11.2 选择题（可多选） …… 430
第 9 章 IP 地址和子网 …… 434
9.1 IPv4 地址 …… 434
9.1.1 IPv4 地址基本格式 …… 435
9.1.2 几个重要概念 …… 436
9.1.3 IPv4 地址分类 …… 437
9.1.4 IPv4 地址前缀表示形式 …… 441

9.1.5 公网/私网 IPv4 地址 …… 442
9.1.6 同种特殊的 IPv4 地址 …… 443
9.2 IPv4 子网划分与聚合 …… 444
9.2.1 IPv4 子网划分的意义 …… 444
9.2.2 VLSM 子网划分的基本思想 …… 447
9.2.3 “全 0 子网”与“全 1 子网” …… 448
9.2.4 广播地址的分类 …… 449
9.2.5 VLSM 子网划分方法 …… 450
9.2.6 CIDR 子网聚合的基本思想 …… 453
9.2.7 子网聚合方法及示例 …… 455
9.2.8 网络地址、广播地址和主机地址的考虑 …… 457
9.3 IPv6 地址基础 …… 458
9.3.1 IPv6 地址表示形式 …… 458
9.3.2 IPv6 地址中的二进制与十六进制转换 …… 460
9.3.3 IPv6 地址类型 …… 461
9.3.4 IPv6 地址前缀表示形式 …… 461
9.4 IPv6 单播地址 …… 462
9.4.1 IPv6 全球单播地址 …… 462
9.4.2 IPv6 本地单播地址 …… 464
9.4.3 内嵌 IPv4 地址的 IPv6 单播地址 …… 465
9.4.4 两种特殊的 IPv6 单播地址 …… 466
9.5 IPv6 组播地址和任播地址 …… 466
9.5.1 IPv6 组播地址 …… 466
9.5.2 IPv6 任播地址 …… 467
9.5.3 IPv6 主机和路由器地址 …… 468
9.6 课后自测题 …… 469
9.6.1 填空题 …… 469
9.6.2 选择题（可多选） …… 469
第 10 章 传输层 …… 476
10.1 传输层基础 …… 477
10.1.1 划分传输层的必要性 …… 477
10.1.2 传输层的端到端传输服务 …… 479
10.1.3 传输层服务 …… 480
10.1.4 TSAP 和 TPDU …… 482
10.1.5 传输连接建立阶段的主要 TPDU …… 485
10.1.6 数据传输阶段的主要 TPDU …… 487
10.1.7 传输连接释放阶段的 TPDU …… 490
10.1.8 传输服务原语 …… 491
10.2 传输层服务功能 …… 495
10.2.1 传输层寻址方案 …… 495
10.2.2 传输连接建立 …… 498
10.2.3 重复传输连接的解决方法 …… 499
10.2.4 数据传输 …… 502
10.2.5 传输连接释放 …… 502
10.2.6 流量控制 …… 504
10.2.7 多路复用 …… 507
10.2.8 崩溃恢复 …… 508
10.3 TCP 协议基础 …… 508
10.3.1 TCP 主要特性 …… 509
10.3.2 TCP 数据段格式 …… 510
10.3.3 TCP 套接字 …… 513
10.3.4 TCP 端口 …… 516
10.3.5 TCP 连接的有限状态机 …… 518
10.3.6 TCP 传输连接的建立 …… 521
10.3.7 TCP 传输连接的释放 …… 525
10.4 TCP 的可靠传输机制 …… 527
10.4.1 TCP 字节编号机制 …… 527
10.4.2 TCP 正确接收确认机制 …… 528
10.4.3 TCP 超时重传机制 …… 529
10.4.4 TCP 选择确认机制 …… 531
10.5 TCP 的流量控制 …… 532
10.5.1 TCP 流量控制 …… 533
10.5.2 传输效率的考虑 …… 534
10.6 TCP 的拥塞控制 …… 535
10.6.1 TCP 拥塞控制概述 …… 536
10.6.2 TCP 拥塞控制的慢启动方案 …… 538
10.6.3 TCP 拥塞控制的拥塞避免方案 …… 539

10.6.4 TCP 拥塞控制的快速重传/快速恢复方案 …… 540
10.7 UDP 协议 …… 541
10.7.1 UDP 协议概述 …… 542
10.7.2 UDP 数据报头部格式 …… 542
10.8 课后自测题 …… 544
10.8.1 填空题 …… 544
10.8.2 选择题（可多选） …… 545
第 11 章 应用层 …… 550
11.1 应用层概述 …… 551
11.1.1 应用层组件及典型应用服务 …… 551
11.1.2 应用层 C/S 服务模型 …… 552
11.2 Web 服务基础 …… 553
11.2.1 Web 服务模型 …… 553
11.2.2 万维网的全球统一标识 …… 554
11.2.3 万维网文档标记 …… 556
11.2.4 HTML 文档类型 …… 558
11.2.5 HTML 文档的“三超属性” …… 559
11.2.6 HTTP 服务访问基本流程 …… 560
11.2.7 HTTP 的主要特性 …… 561
11.2.8 HTTP 请求报文格式 …… 562
11.2.9 HTTP 响应报文格式 …… 565
11.3 DNS 服务 …… 567
11.3.1 DNS 技术的引入背景 …… 567
11.3.2 DNS 命名方案的设计思想 …… 569
11.3.3 DNS 名称空间 …… 571
11.3.4 DNS 名称服务器 …… 573
11.3.5 DNS 报文格式 …… 577
11.3.6 DNS 数据传输方式 …… 581
11.3.7 DNS 递归解析原理 …… 582
11.3.8 DNS 迭代解析原理 …… 585
11.4 DHCP 服务 …… 587
11.4.1 BOOTP 和 DHCP 简介 …… 587
11.4.2 DHCP 服务的主要功能及应用环境 …… 588
11.4.3 DHCP 报文及其格式 …… 590
11.4.4 DHCP 服务 IP 地址自动分配原理 …… 593
11.4.5 DHCP 服务 IP 地址租约更新原理 …… 600
11.4.6 DHCP 中继代理服务 …… 600
11.5 电子邮件服务 …… 604
11.5.1 电子邮件系统的基本结构 …… 604
11.5.2 电子邮件消息格式 …… 606
11.5.3 SMTP 请求命令和应答消息 …… 608
11.5.4 SMTP 服务工作原理 …… 613
11.5.5 POP3 请求命令及应答消息 …… 615
11.5.6 POP3 服务工作原理 …… 618
11.5.7 IMAP4 协议简介 …… 620
11.6 FTP 服务 …… 621
11.6.1 FTP 服务体系结构及传输方式 …… 621
11.6.2 FTP 的两种工作模式 …… 622
11.6.3 FTP 协议数据类型 …… 628
11.6.4 FTP 协议格式控制 …… 630
11.6.5 FTP 协议数据结构 …… 630
11.6.6 FTP 协议传输模式 …… 631
11.6.7 FTP 协议操作命令 …… 633
11.6.8 FTP 协议响应码 …… 635
11.6.9 FTP 协议身份验证 …… 637
11.7 SNMP 服务 …… 638
11.7.1 SNMP 协议的主要版本 …… 638
11.7.2 SNMP 协议管理模型 …… 640
11.7.3 SNMPv1 报文格式 …… 642
11.7.4 SNMPv2c 报文格式 …… 645
11.7.5 SNMPv3 报文格式 …… 647
11.7.6 SNMPv1 和 SNMPv2c 实现机制 …… 649
11.7.7 SNMPv3 实现机制 …… 651
11.8 课后自测题 …… 652
11.8.1 填空题 …… 652
11.8.2 选择题（可多选） …… 653
附录 各章课后自测题参考答案 …… 659

1 数制基础和机器数运算

“数制”是“数据进制”或“数据制式”的简称，也就是数据逢几进一的意思，如我们常用的十进制就是逢十进位。当然，数制的类型远不只十进制，在计算机系统中常见的还有二进制、八进制和十六进制这三种。

本章主要介绍以上各种不同类型数制的特点及它们之间的相互转换方法、在计算机中参与运算的机器数类型（定点数和浮点数）、机器数的各种表示形式（原码、反码、补码、阶码和移码），以及它们的算术/逻辑运算方法。

本章内容是为我们学习本书以后各章涉及数制方面的内容打基础的，如后面章节中将要介绍的各种信息编码方式、IPv4/IPv6 地址格式转换、IPv4 网络地址的计算、IPv4 子网划分与聚合、MAC 地址格式转换等。另外，本章所讲内容对于我们在日常故障排除中所进行的数据包分析也是非常必要的。当然，如果你对这些内容已有掌握，可直接跳过。

本章重点与难点：

- 十进制数与非十进制数间的相互转换（要区分整数和小数部分的不同转换方法）
- 机器数的五种编码方式（原码、反码、补码、阶码和移码）
- 机器数的表示形式及存储结构
- IEEE 754 标准浮点数的计算
- 二进制数的算术、逻辑运算方法
- 补码和浮点数的加、减法运算方法

1.1 数制概述

“数制”是“数据进制”的简称，是指数据的进位计数规则，又称为“进位计数制”，简称“进制”。本节先来简单地了解一些常见的数制类型及其特点。

1.1.1 常见数制类型及表示方法

日常生活中我们经常使用的数是十进制的，如我们拿的 3000 元工资、市场 1.5 元/斤的菜价等。之所以称其为十进制，是因为这类数是逢十进一的。除了十进制计数以外，还有许多其他进制的计数方法。在计算机中常见的有二进制、八进制、十六进制等制式，这三种进制的数在进行加法运算中分别是逢二、八、十六进一，这就是前面所说的进位计数规则。关于如何理解这些不同数制类型数据的加法运算，在本章后面将有专门介绍。

其实数制类型远不止这么几种，如我们以 60 分钟为 1 小时，60 秒为 1 分钟，用的就是六十进制计数法；一天之中有 24 小时，用的是二十四进制计数法；一星期有 7 天，用的是七进制计数法。

虽然数据制式可以有很多种，但在计算机通信中通常遇到的仍是以上提到的二进制、八进制、十进制和十六进制这 4 种。在一种数制中所能使用的数码的个数称为该数制的“基数”，也就对应数制类型的名称，如二进制的基数为“2”，八进制的基数为“8”，十进制的基数为“10”，十六进制的基数也就是“16”。这里所说的“基数”其实就是前面所说的进位计算规则，如我们常见的十进制数是逢十进一，二进制数是逢二进一，……。

既然有不同的数制，那么在计算机程序中给出一个数时就必须指明它属于哪一种数制，否则计算机程序就不知道该把它看成是哪种数了（当然，在计算机及其他计算机网络设备内部都是以二进制进行运算的）。如 12300 这个数，既可能是十进制又可能说是八进制或者十六进制，所以数需要有专门的标志来进行区别。

（1）十进制（Decimal）。

十进制是日常生活中常用的数制类型，基数是 10，也就是它有 10 个数字符号，即 0、1、2、3、4、5、6、7、8、9。其中最大数码是“基数”减 1，即 10-1=9，最小数码是 0。十进制数的标志为 D，如(1250)D，也可用下标“10”来表示，如$(1250)_{10}$（注意是下标）。也可以不加这些标志，默认就是十进制。

（2）二进制（Binary）。

二进制是计算机运算时所采用的数制，基数是 2，也就是说它只有两个数字符号，即 0 和 1。如果在给定的一个数的表示形式中，除 0 和 1 外还有其他数（如 1061），那它就绝不会是一个二进制数了。二进制数的最大数码也是基数减 1，即 2-1=1，最小数码也是 0。二进制数的标志为 B，如(1001010)B，也可用下标“2”来表示，如$(1001010)_2$（注意是下标）。

（3）八进制（Octal）。

八进制的基数是 8，也就是说它有 8 个数字符号，即 0、1、2、3、4、5、6、7。对比十进制可

以看出，它比十进制少了两个数“8”和“9”，这样当一个数的表示形式中出现“8”和（或）“9”时（如23459），那它也就绝不是八进制数了。八进制数的最大数码也是基数减1，即8-1=7，最小数码也是0。八进制数的标志为O或Q（注意它特别一些，可以有两种标志），如(4603)O（注意是字母O，不是数字0）、(4603)Q，也可用下标“8”来表示，如$(4603)_8$（注意是下标）。在C、C++这类编程语言中规定，一个数如果要指明它采用八进制，必须在它前面加上一个0，如123是十进制，但0123则表示采用八进制。

（4）十六进制（Hexadecilnal）。

十六进制数用得比较少，最新的IPv6地址就是采用十六进制来表示的（IPv4地址通常采用的是十进制表示）。在注册表中也会用到十六进制，所以了解十六进制还是非常必要的。

十六进制的基数是16，也就是说它有16个数码，除了十进制中的10个数码可用外，还使用了6个英文字母（分别代表10、11、12、13、14、15），这样一来，十六进制的这16个数码依次是0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F（不区分大小写）。对比前面对其他几种数制的介绍可以看出：如果一个数的表示式中出现了字母，如63AB，则一定不会是二进制、八进制或十进制，而是十六进制了。

十六进制的最大数码也是基数减1，即16-1=15（为F），最小数码也是0。十六进制数的标志为H，如(4603)H，也可用下标“16”来表示，如$(4603)_{16}$（注意是下标）。十六进制数也常用前缀0x来表示（注意是数字0，而不是字母O）。在C、C++这类编程语言中也规定，十六进制数必须以0x开头。比如0x10表示一个十六进制数，而不是八进制或者十进制的10。



既然在计算机中使用的是二进制，那为什么还要十进制、八进制和十六进制呢？其实这都不是计算机自身要求的，因为在计算机运算中全都是使用二进制。之所以还需要这些数制，完全是出于表达和识别的方便性考虑的。

因为一个较大的数用二进制表示的话就太长了，如一个C、C++等编程语言中的int（整数）类型的数据要占用4个字节，也就是32位。比如100，用int类型的二进制数表达将是：0000 0000 0000 0000 0110 0100。这还是一个比较小的数，如果数更大，则会更复杂。试想一下，要写这么长，估计没几个人会喜欢，于是就有了可以更简便表示的十进制、八进制和十六进制，因为数制越大，表示一个数所需的数码位数就越少。所以，像C、C++这类语言没有提供在代码中直接输入二进制数的方法，而是普遍采用八进制或十六进制。

那为什么不是其他进制类型，如九进制或二十进制呢？原因就在于2、8、16分别是2的1次方、3次方、4次方，这就使得这三种进制之间可以非常直接地互相转换。八进制或十六进制缩短了数的表示位数，但保持了二进制数的表达特点。在下面关于进制转换的介绍中你就可以发现这一点。

以上4种进制的基本特点比较如表1-1所示。

表 1-1　4 种进制的基本特点比较

数制	基数（数码个数）	数码	表示方式
二进制	2	0、1	数后面加 B 或下标 2
八进制	8	0、1、2、3、4、5、6、7	数后面加 O 或 Q 或下标 8
十进制	10	0、1、2、3、4、5、6、7、8、9	数后面加 D 或下标 10，也可以不加
十六进制	16	0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F	数后面加 H 或下标 16

1.1.2　不同数制之间的对应关系

表 1-2 所示是以上介绍的二进制、十进制、八进制和十六进制这 4 种常在计算机中使用的数制的对应关系。注意，因为八进制没有 8 和 9，所以二进制 1000 对应的八进制是 10，而不是想象中的 8，二进制 1001 对应的八进制是 11，而不是想象中的 9。这个表很重要，大家最好全部记下来，特别是这几种数制的对应关系，这样可以方便以后不同数制间转换时的计算。

表 1-2　不同数制的对应关系

二进制数	对应的十进制数	对应的八进制数	对应的十六进制数
0	0	0	0
1	1	1	1
10	2	2	2
11	3	3	3
100	4	4	4
101	5	5	5
110	6	6	6
111	7	7	7
1000	8	10	8
1001	9	11	9
1010	10	12	A
1011	11	13	B
1100	12	14	C
1101	13	15	D
1110	14	16	E
1111	15	17	F

1.2 不同数制间的相互转换

同一个数在一些环境中（如要进行子网划分、计算子网掩码时、对信息进行编码时等），可能要用不同数制形式来表示，这就涉及到数制间的转换问题了。下面是常见的十进制、二进制、八进制、十六进制之间的转换方法。

1.2.1 非十进制数转换成十进制数

非十进制转换成十进制就是把这些非十进制按位以对应的权值（注意要区分整数位和小数位）展开，然后相加即得出相应的十进制值。本节后面介绍的各种非十进制转换成十进制的方法都是按照这种方法进行的。

“权值”是指对应数值位的进制幂次方数，如二进制整数中第 0 位（最低位，也就是整数最右边的那位）的权值是 2 的 0 次方，第 1 位的权值是 2 的 1 次方，……。同理在八进制整数中第 0 位的权值是 8 的 0 次方，第 1 位的权值是 8 的 1 次方，……，依此类推。但每位的权值会因是整数位还是小数位而不同：

- 整数的第 0 位（也就是最低位）的权值为对应进制的 0 次方，最高位的权值为对应进制的 n-1 次方。
- 小数的第一位（最靠近小数点的那位，也是小数的最高位）的权值为对应进制的-1 次方，最后一位（最右边的那位，亦即小数的最低位）的权值为对应进制的-n 次方。

1. 二进制转换为十进制

二进制转换成十进制的方法大家可能早就有所了解了，如在 IPv4 地址计算时就经常进行这样的操作。转换的方法比较简单，只需按它的权值展开即可。展开的方式是把二进制数首先写成加权系数展开格式，然后按十进制加法规则求和。这种做法称为“按权相加”法。

二进制整数部分的一般表现形式为：$b_{n-1}\ldots b_1b_0$（共 n 位），按权相加展开后的展开格式为（注意，展开式中从左往右各项的幂次是从高到低下降的，最高位的幂为 n-1，最低位的幂为 0）：

$$b_{n-1}\times2^{n-1}+b_{n-2}\times2^{n-2}+\ldots+b_1\times2^1+b_0\times2^0$$

【示例 1】二进制数$(11010)_2$的按权相加展开格式为：

$$1\times2^4+1\times2^3+0\times2^2+1\times2^1+0\times2^0=16+8+0+2+0=(26)_{10}$$

二进制小数部分的幂次是反序排列的（也就是与整数部分的幂次序列相反，从右往左其绝对值是从低到高上升的），且为负值，最高位幂次（也就是最靠近小数点的第一个小数位的幂次）为“-1”。如二进制小数部分的格式为：$0.b_{n-1}\ldots b_1b_0$，则按权相加后的展开格式为：

$$b_{n-1}\times2^{-1}+b_{n-1}\times2^{-2}+\ldots+b_1\times2^{-(n-1)}+b_0\times2^{-n}$$

【示例 2】$(0.1011)_2$的按权相加展开格式为：

$$1\times2^{-1}+0\times2^{-2}+1\times2^{-3}+1\times2^{-4}=0.5+0+0.125+0.0625=(0.6875)_{10}$$

2. 八进制转换为十进制

八进制转换成十进制也是采取“按权相加”法，只是这里的权值是 8 的相应幂次方。如八进制整数部分的格式为：$b_{n-1}\ldots b_1b_0$，则按权值相加展开后的格式就为（从左往右幂次是从高到低下降的）：

$$b_{n-1}\times8^{n-1}+b_{n-2}\times8^{n-2}+\ldots+b_1\times8^1+b_0\times8^0$$

【示例 3】八进制数$(26356)_8$的按权值相加展开格式为：

$$2\times8^4+6\times8^3+3\times8^2+5\times8^1+6\times8^0=8192+3072+192+40+6=(11502)_{10}$$

八进制小数部分的幂次也是反序排列的（也就是与整数部分的幂次序列相反，从右往左其绝对值是从低到高上升的），且为负值，最低幂次（也就是最靠近小数点的第一个小数位的幂次）为“-1”。如八进制小数部分的格式为：$0.b_{n-1}\ldots b_1b_0$，则按权相加后的展开格式为：

$$b_{n-1}\times8^{-1}+b_{n-2}\times8^{-2}+\ldots+b_1\times8^{-(n-1)}+b_0\times8^{-n}$$

【示例 4】$(0.257)_8$按权相加的展开格式为：

$$2\times8^{-1}+5\times8^{-2}+7\times8^{-3}=0.25+0.078125+0.013671875=(0.341796875)_{10}$$

3. 十六进制转换为十进制

十六进制转换成十进制的方法也是采取“按权相加”法，只是这里的权值是 16 的相应幂次方。如十六进制整数部分的格式为：$b_{n-1}b_{n-2}\ldots b_1b_0$，则按权相加展开后的格式就为（从左往右幂次是从高到低下降的）：

$$b_{n-1}\times16^{n-1}+b_{n-2}\times16^{n-2}+\ldots+b_1\times16^1+b_0\times16^0$$

【示例 5】十六进制数$(26345)_{16}$的按权相加展开后的格式为：

$$2\times16^4+6\times16^3+3\times16^2+4\times16^1+5\times16^0=131072+24576+768+64+5=(156485)_{10}$$

十六进制小数部分的幂次也是反序排列的（也就是与整数部分的幂次序列相反，从右往左其绝对值是从低到高上升的），且为负值，最低幂次（也就是最靠近小数点的第一个小数位的幂次）为“-1”。如十六进制小数部分的格式为：$0.b_{n-1}\ldots b_1b_0$，则按权相加后的展开格式为：

$$b_{n-1}\times16^{-1}+b_{n-2}\times16^{-2}+\ldots+b_1\times16^{-(n-1)}+b_0\times16^{-n}$$

【示例 6】$(0.25A)_{16}$按权值相加展开后的格式为：

$$2\times16^{-1}+5\times16^{-2}+10\times16^{-3}=0.125+0.01953125+0.00244140625=(0.14697265625)_{10}$$

1.2.2 十进制数转换成非十进制数

十进制数转换成非十进制数的方法是：整数部分的转换用“除基取余法”，也就是用基数相除，然后从最后的商开始反序（由后向前）取余数；小数部分的转换用“乘基取整法”，也就是用基数相乘，然后正序（由前向后）取得到的整数。这里的“基数”就是对应的数制，如二进制的基数为 2，八进制的基数为 8，十六进制的基数为 16。

1. 十进制转换为二进制

这里分别对十进制整数和十进制小数转换成二进制进行介绍。

（1）十进制整数转换成二进制的方法。

十进制整数转换为二进制的方法是采用“除 2 逆序取余”法（采用短除法进行）。也就是先将十进制数除以 2，得到一个商数（也是下一步的被除数）和余数；然后再将商数除以 2，又得到一个商数和余数；依此类推，直到商数为小于 2 的数为止。然后从最后一步得到的小于 2 的商开始将其他各步所得的余数（也都是小于 2 的 0 或 1）排列起来（俗称“逆序排列”）就得到了对应的二进制数。

【注意】这里与下面的小数转换有些不一样，这里要包括最后得到的小于 2 的商数，而小数转换中是不需要包括最后的积的，只包括各步得到的整数部分，后面的十进制整数转换为八进制、十六进制也一样。

【示例 1】图 1-1 所示为十进制整数 48 转换成二进制数时依次除 2 的过程。

在每步的最右边显示的是各步商数除 2 所得到的余数，最后一步的商数为 1，因为它小于 2，所以不能再除了。然后从最后得到的商数（1）开始依次向上把其他各步除 2 得到的余数排列起来，就得到最后 48 转换成二进制时的结果为$(110000)_2$。同理，图 1-2 所示的十进制数 250 转换成二进制数后的结果就为$(11111010)_2$。

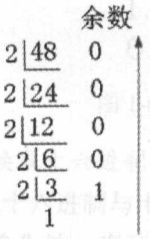


图 1-1 十进制整数 48 转换成二进制整数的步骤

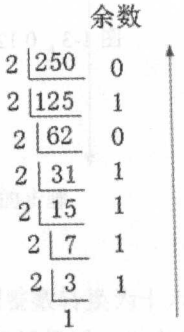


图 1-2 十进制整数 250 转换成二进制整数的步骤

（2）十进制小数转换成二进制的方法。

十进制小数转换为二进制的方法是采用“乘 2 正序取整”法。也就是用 2 乘十进制小数，得到一个积，然后将积的整数部分取出作为相应步骤得到的整数；再用 2 乘余下的小数部分，又得到一个积，然后再将这个积的整数部分取出；依此类推，直到积中的小数部分为零，或者达到所要求的精度为止；最后把各步取出的整数部分（仅需要各步得到的整数部分，而不需要最后没有取整的小数部分）按正序排列起来，即先取的整数作为二进制小数的高位，后取的整数作为低位。

【示例 2】图 1-3 中左、右图所示的是分别将十进制小数 0.125 和 0.625 转换成二进制的过程。

最后得到的二进制数就是从最开始得到的整数值开始一直到最后得到的整数值（也就是自上而下的顺序，与整数转换中取余的顺序相反）。0.125 和 0.625 最后的二进制值分别为$(0.001)_2$和$(0.101)_2$（注意，一定要记得在整数部分加上“0.”，因为十进制小数转换成二进制后仍是小数）。

【说明】有些十进制小数乘以 2 后是个无穷循环数，永远不会有完整的整数，此时就要看所需的精度如何了，按所需位数精度取值即可。如 0.825 就是这样一个数，如果仅要求是小数点后 3

位，则相应的二进制数为$(0.110)_2$，如果要求为4位，则对应的二进制值为$(0.1101)_2$，具体如图1-4所示。

如果一个十进制同时有整数和小数部分，则要对整数和小数部分分别按以上介绍的对应方法进行二进制转换。

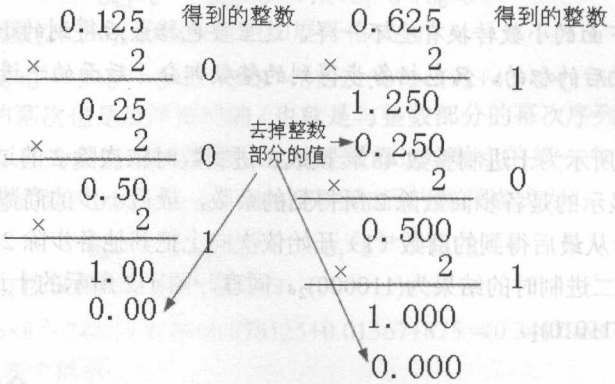


图1-3　0.125和0.625十进制小数转换成二进制小数的步骤

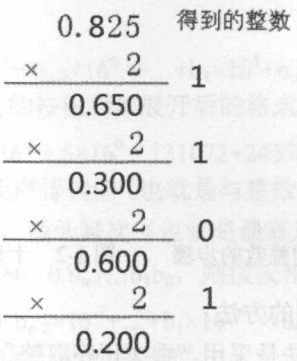


图1-4　不同精度要求的取值示例

2. 十进制转换成八进制

十进制转换成八进制的方法与前面介绍的十进制转换成二进制的方法类似，只不过这里的基数是8（而不再是2）。十进制转换成八进制当然也分整数部分和小数部分两种不同的转换方法。

十进制整数转换为八进制整数采用“除8逆序取余”法，直到所得的商小于8，然后把余数（包括最后一步中得到的小于8的商）按逆序排列即可；十进制小数转换为八进制小数采用“乘8正序取整”法，直到所得到的积小数部分为0或者在规定的精度范围内，然后把所得到的整数正序排列起来即可。

【示例3】图1-5中左、右图所示的是分别将十进制整数65和2467按“除8逆序取余”法转换成八进制的步骤，得到的结果分别是$(101)_8$和$(4643)_8$。

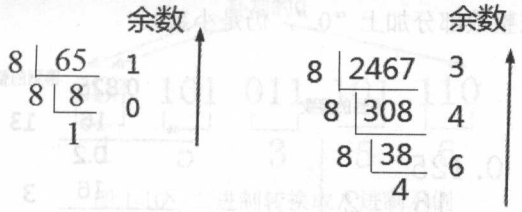


图 1-5 两个十进制整数转换成八进制的步骤

【示例 4】图 1-6 中左、右图所示的是两个十进制小数 0.125 和 0.8125 通过“乘 8 正序取整”法转换成八进制的步骤，得到的结果分别是$(0.1)_8$和$(0.64)_8$（是正序排列，一定要记得在整数部分加上“0.”）。

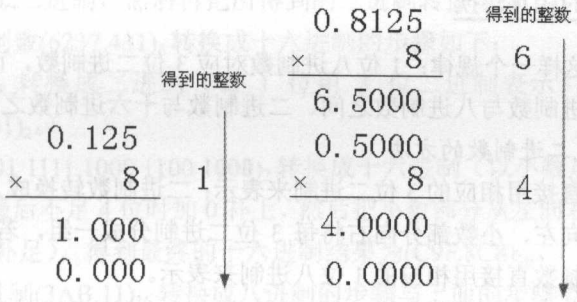


图 1-6 两个十进制小数转换成八进制小数的步骤

3. 十进制转换成十六进制

十进制转换成十六进制与十进制转换成二进制类似，将十进制整数转换为十六进制的方法是采用“除 16 逆向取余”法，直到所得的商小于 16，然后把余数（包括最后一步中得到的小于 16 的商）按逆序排列即可；十进制小数转换为十六进制的方法是采用“乘 16 正序取整”法，直到所得到的积小数部分为 0 或者在规定的精度范围内，然后把所得到的整数正序排列起来。

【示例 5】图 1-7 左、右图所示的是分别将十进制整数 45 和 3456 按“除 16 逆序取余”法转换成十六进制的步骤，得到的结果分别是$(2D)_{16}$和$(D80)_{16}$（注意，其中的 13 用十六进制的 D 表示了）。

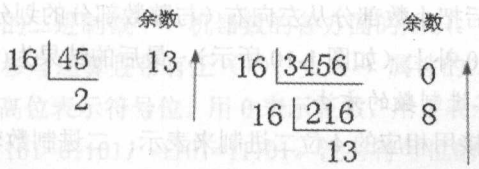


图 1-7 两个十进制整数转换成十六进制的步骤

【示例 6】图 1-8 中左、右图所示的是将十进制小数 0.125 和 0.825 通过“乘 16 正序取整”法转换成十六进制的步骤，得到的结果分别是$(0.2)_{16}$和$(0.D33)_{16}$（精确到小数点后面三位）（注意：是

正序排列，也一定要记得在整数部分加上“0.”，仍是小数）。

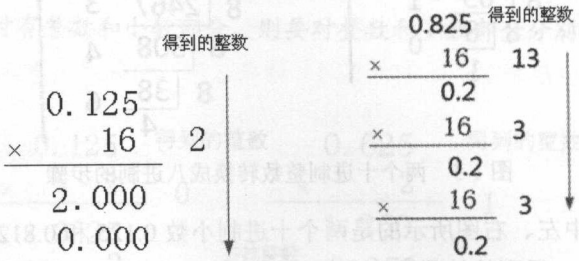


图 1-8　两个十进制小数转换成十六进制小数的步骤

1.2.3　非十进制数之间的相互转换

从表 1-2 可以得出这样一个规律：1 位八进制数对应 3 位二进制数，而 1 位十六进制数对应 4 位二进制数。因此，二进制数与八进制数之间、二进制数与十六进制数之间的相互转换十分容易。

1. 八进制数转换成二进制数的方法

将每 1 位八进制数直接用相应的 3 位二进制来表示；二进制数转换成八进制数的方法是：以小数点为边界，整数部分向左，小数部分向右将每 3 位二进制分成一组，若不足 3 位则用 0 补足 3 位；然后将每一组二进制数直接用相应的 1 位八进制来表示。

【示例 1】将$(3456.2262)_8$转换为二进制数的方法是依次把八进制的每 1 位用 3 位二进制表示（如图 1-9 所示），最后的结果为$(11100101110.010010110010)_2$（整数部分最前面的 0 可以省略）。

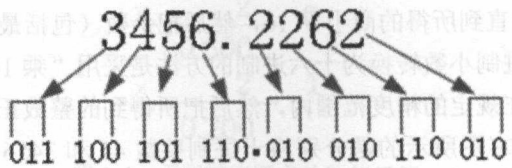


图 1-9　八进制转换成二进制示例

【示例 2】将$(1101011.10111)_2$转换为八进制数，可把整数部分从右向左每 3 位分为一组，最后不足 3 位时加 0 补上，然后把小数部分从左向右（与整数部分的划分顺序相反）同样以每 3 位分为一组，最后不足 3 位时加 0 补上（如图 1-10 所示），最后的结果为$(153.56)_8$。

2. 十六进制数转换成二进制数的方法

将每 1 位十六进制数直接用相应的 4 位二进制来表示；二进制数转换成十六进制数的方法是：以小数点为边界，整数部分向左，小数部分向右将每 4 位二进制数分成一组，若不足 4 位则用 0 补足 4 位；然后将每一组二进制数直接用相应的 1 位十六进制表示。

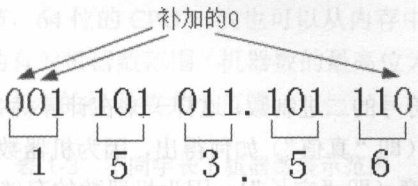


图 1-10 二进制转换成八进制示例

【示例 3】将$(4AF.51)_{16}$转换成二进制数的最后结果为$(100\ 1010\ 1111.0101\ 0001)_2$；将二进制数$(11\ 0110\ 1110.\ 1010\ 1010\ \mathbf{1000})_2$（最后加粗部分的 0 是划分位时补上去的，下同）转换为十六进制数的最后结果为$(36E.AA8)_{16}$。

3. 八进制与十六进制的相互转换方法

先把其中一个转换成二进制，然后再把所得到的二进制转换成另一个进制的数。

【示例 4】将八进制数$(6237.431)_8$转换成十六进制的步骤如下：

（1）将$(6237.431)_8$转换成二进制（每 1 位用 3 位二进制表示），得到的二进制结果是$(110010011111.100011001)_2$。

（2）再将$(1100\ 1001\ 1111.1000\ 1100\ \mathbf{1000})_2$转换成十六进制（以小数点为边界，整数部分从右向左每 4 位分为一组，最后不足 4 位时加 0 补上，然后把小数部分从左向右将每 4 位二进制数分成一组，不足 4 位则用 0 补足），得到最终的十六进制结果为$(C9F.8C8)_{16}$。

【示例 5】将十六进制$(3AB.11)_{16}$转换成八进制的步骤与上面的步骤类似，具体转换过程如下：

（1）先将十六进制数$(3AB.11)_{16}$转换成二进制，得到$(001110101011.00010001)_2$。

（2）再将$(001\ 110\ 101\ 011.000\ 100\ \mathbf{010})_2$转换成八进制，以小数点为边界，整数部分从右向左每 3 位分为一组，最后不足 3 位时加 0 补上，然后把小数部分从左向右将每 3 位二进制数分成一组，不足 3 位则用 0 补足，最后的结果为$(1653.042)_8$。

以上二进制、八进制、十六进制的对应关系可参见表 1-2，根据该表可直接进行转换。

1.3 机器数基础

在本节之前介绍的数制之间的转换没有考虑二进制数是带有符号位的情况，当然这不影响我们对前面介绍的数制间转换方法的学习，因为数制之间的转换是不包括符号位的，即符号位保持不变。本节我们专门来介绍带符号的二进制数——机器数的各方面的知识。

“机器数”是计算机中参与运算且带有正（+）、负（-）属性的二进制数。即机器数是有符号位的二进制数，并且规定最高位表示符号位，用 0 表示正数，用 1 表示负数，所以机器数也称为符号数值化的二进制数。如+1101=01101，-1101=11101。没有符号位的二进制数称为“无符号数”，也不是机器数，不参与各种运算，如各种信息编码、ID 编号等。

1.3.1 机器数的真值

前面说了，机器数是带有符号的二进制数，这样一来在计算机中参与运算的机器数涉及到两个问题：一是机器数的真实数值（即“真值”）如何得出，因为机器数中的符号位也是用 1、0 来表示的；二是存储器的单位存储长度（即“字长”），因为机器数的存储和读取是受到计算机中 CPU 一次可以处理的数据位数或者内存中用于存储数据的基本单位限制的。本节先介绍什么是机器数的“真值”。

在机器数中规定最高位用来表示数据符号，其中 1 代表为负，0 代表为正。这样一来，存储在内存中的机器数本身就不能代表对应数据的真正数值了，只有去掉最高的符号位后才是数值本身，这就是机器数的“真值”。

【示例】机器数 10000101 的最高位 1 代表负（-），所以余下来的“0000101”才是数值本身，所以其真正数值（即“真值”）是-5。如果是无符号数，则 10000101 所代表的是 133。

为了区别，带符号位的机器数对应的真正数值称机器数的“真值”。例如，00100001 的真值 = 0 0100001 = +33（正号可以不写，可以直接写成 33），10100011 的真值 = 1 0100011 = -35。

1.3.2 机器数的字长

“字长”可以简单理解为用来表示一个机器数所用的二进制位数。如果确定了字长，则每个机器数都是用固定的二进制位数来表示，不管这个机器数的大与小。但字长不同，同一个机器数的表示形式也不一样。如果字长为 8 位，十进制中的数+5 转换成二进制就是 00000101，-5 转换成二进制就是 10000101；但如果字长是 16 位，+5 转换的结果就是 00000000 00000101，而-5 转换成二进制就是 10000000 00000101 了。也就是对应的机器数要转换为字长所代表的位数，因为一个机器数必须单独占用一个字长（非机器数没这个要求，由其对应的编码方式决定），否则计算机无法知道一个存储单元中的机器数到底是多少。

【注意】“-0”与“0”的机器数在内存中的存储是不一样的，在 8 位字长中，-0 为 1 0000000，而+0 为 0 0000000；在 16 位字长中，-0 为 1 0000000 00000000，而+0 为 0 0000000 00000000。所以在二进制的机器数中，0 也有两个（-0 和 0），且表示形式并不一样。

另外，字长的选定也不是随意的，它与我们通常所说的 16 位、32 位或 64 位 CPU、内存等硬件体系架构是一一对应的。这里的 16 位、32 位、64 就对应了在相应的计算机系统中的字长。计算机内存中数据的存储是以单位长度进行连续存储的，一个机器数只能在一个存储单元中存储，一个存储单元也只能存储一个机器数，因为计算机中的 CPU 也是以存储单元为单位从内在中读取并处理数据的。所以，“字长”也是指计算机一次可处理的机器数的码位长度，是计算机进行数据存储和数据处理的运算单位。如我们通常所指的 32 位处理器就是指该处理器的字长为 32 位，也就是一次能处理最大数值是 32 位的数。

显然，字长越长，CPU 的数据处理能力越强，因为它一次可以处理的数据越多。我们知道，8 位是一个字节，16 位就是两个字节。现在的计算机设备基本上都是 64 位字长的，也就是在内存的

一个存储单元中可以存 8 个字节，64 位的 CPU 一次也可以从内存中读取并处理 8 个字节。当然，这也决定了不同字长所能表示的有效机器数范围（机器数的最高位为符号位，不参与计算）是不同的，表 1-3 列出了主要字长所能表示的机器数大小范围。

表 1-3 不同字长下机器数表示范围

字长	真值位长	机器数表示范围
8 位	7	$-2^7-1 \leqslant +2^7-1$，即-127≤+127
16 位	15	$-2^{15}-1 \leqslant +2^{15}-1$，即-32767≤+32767
32 位	31	$-2^{31}-1 \leqslant +2^{31}-1$
64 位	63	$-2^{63}-1 \leqslant +2^{63}-1$

1.4 机器数的编码形式

为了使机器数的运算方法适用于所有机器数，而且运算不会出现二义性，在机器数的运算方法设计过程中出现了 3 种不同的机器数编码方式：原码、反码、补码。另外，在本章后面将要介绍的浮点机器数运算中又引入了“阶码”和“移码”这两个概念。下面对这 5 种表示编码进行具体介绍。

1.4.1 原码

对于人脑来说，我们都知道，+表示正数，-表示负数，但在计算机二进制中也引入这两个符号肯定是不行的，因为在计算机中只有 0 和 1 这两个数字，根本不认识“+”和“-”这两个符号。计算机中的任何行为都依赖它的物理结构，它没有思维，所以你得在 0 和 1 之间让计算机识别出对应数的正与负。

1. 原码的编码规则

“原码”就是“原始码位”或“原始编码”的意思，就是对应机器数本身所代表的编码形式。即机器数的最高位代表符号，1 表示负，0 表示正，其余各位为真值位。

【注意】一个机器数的原码表示形式是与字长有关的，这在 1.3.2 节已有介绍。当指定了字长时，如果原来的机器数真值二进制码长度+1（符号也要占一位）小于字长时，则包括了整数的机器数，要在真值（除最高位为符号位外，其余各位均为真值位）最前面（最左边）加上 0 来补足；纯小数的机器数，要在真值最后面（最右边）加 0 补足。

比如，+3 的符号位为 0，真值 3 转化为二进制就是 11，如果不考虑字长的话，则+3 的原码就是 011；但如果指定字长为 8，则+3 的原码就是 **00000011** 了（加粗部分为补的 5 个 0）。同理，-3 的符号位为 1，真值 3 转化为二进制就是 11，如果不考虑字长的话，-3 的原码是 111；但如果指定字长为 8，则-3 的原码就是 **10000011**（加粗部分为补的 5 个 0）。如 0.11010 的机器数，指定字长为 8，则其原码为 0.1101**000**（加粗部分为补的 2 个 0，小数点是不计数位的）。

在日常的书写中，原码的表示形式是用方括号下面加上一个“原”字下标来区别的，如$[+3]_{原}$

$=00000011$，$[-3]_{原}=10000011$。

2. 原码的优点与缺点

原码的优点是设计简单、容易理解。原码的缺点有如下几个：

- 加减运算规则复杂（符号位要进行单独处理），还可能出错。在采用原码直接进行机器数运算时，同为正数的加、减法运算是没什么问题的，可是异号相加、减时就存在问题了。如 1-1，如果用原码计算的话则如下所示，结果为-2，显然不正确：

$$(0\ 0000001)_{原}+(1\ 0000001)_{原}=(1\ 0000010)_{原}=-2$$

- 原码中的 0 有两种编码方式，存在二义性。如果是 8 位字长，则 0 有 000000000 和 10000000，即有+0 和-0 之分，有二义性。所以直接采用原码进行机器数运算不合适。

1.4.2 补码

正因为原码在机器数运算过程中存在以上不足，所以设计者又继续寻找其他更好的编码方式。最后终于找到了一种能解决原码的那些不足的另一种编码方式——补码。很显然，补码不可能再采用机器数的本身编码方式（即原码）来进行了，需要采用新的编码方式。

1. 补码的编码规则

补码的编码规则为：正数的补码和原码相同；负数的补码是通过先把除符号位外（使数的符号位不变，正、负属性保持不变）其他各位取反（就是求下面将要介绍的该数的“反码”），再在末位（最低位）加 1 得到的。这样，我们只要让减数通过一个求反电路，再通过一个+1 电路，然后再通过加法器就可以实现减法运算了。

如一原码为 01001 的机器数，因为最高位为 0，所以是正数，则它的补码与原码一样为 01001。如另一个原码为 11001 的机器数，因为最高位为 1，所以是负数，先对原码除符号位外的其他各位取反，得到 10110，再在结果的最低位加 1，得到 10111，这就是机器数 11001 的补码了。

2. 补码的优点

补码继承了部分原码的特点（可以表示正与负），并且它有两个新的优点：

- 可以把符号位一起运算，符号位不需要单独处理。

【示例 1】同样以前面的 1-1 为例，如果采用补码形式，则算式如下，结果完全正确：

$$(0\ 0000001)_{补}+(1\ 1111111)_{补}=0\ 0000000=0$$

- 0 只有一种表示形式，没有二义性。

+0 的补码就是其原码本身，字长为 8 位时为 00000000，-0 的原码为 10000000，其补码是先对低 7 位取反，得到 11111111，再在最低位加 1，如仍是 8 位字长的话，结果为 00000000（最高位进的“1”被溢出了），这就是-0 的补码，也和+0 的补码一样。

3. 补码转换成原码的方法

前面介绍了原码转换成补码的方法，如果要把一个补码转换成原码该如何操作呢？很简单，只需要把原码转换成补码的过程倒过来、反向操作即可。

因为正数的补码与原码一样，所以正数补码的原码就是其补码，而负数补码的原码是先在最后

一位减 1（具体的减法运算方法在本章后面介绍），然后对其除符号位外的其他各位全部取反得到。所以关键还是负数补码与原码之间的转换问题。

【示例 2】如已知一个补码为 100110，要求它的原码。按照前面介绍的方法负数的补码是先在补码的最后一位减 1（如图 1-11 所示），得到 100101；然后对以上结果除符号位外的其他各位取反，得到 111010，这就是补码 100110 的原码。

下面再对以上示例通过前面介绍的由原码转换成补码的方法来验证以上由补码求原码的方法的正确性。把前面计算出的原码 111010 转换成补码后是否为 100110。

原码 111010 的最高位为 1，为负数，先对除符号位外的其他各位取反，得到 100101，然后再在最低位加 1（如图 1-12 所示），得到 100110，恰好与最初的补码一致。证明前面介绍的由补码求原码的方法是正确的。

```
  100110
-     1
--------
  100101
```

图 1-11　100100-1 的运算

```
  100101
+     1
--------
  100110
```

图 1-12　100101+1 的运算

【注意】只有相同码制的数才能进行操作，结果就是对应的码制，也就是原码数与原码数的运算结果也为原码，反码数与反码数的运算结果也为反码，补码数与补码数的运算结果也为补码。如果结果是负数，要判断结果是否正确，需要再将其对应的码制转换为原码。

1.4.3　反码

通过以上介绍我们可以知道，补码是我们的最佳选择，因为它的符号位与真值位一样，都参与运算，可以全面地解决异号二进数之间的加减运算问题。那为什么还有“反码”这种表示形式呢？

其实“反码”是“原码”向“补码”表示形式转变过程中的一个过渡形式，不是可直接用于机器数运算的一种编码方式。之所以当初会想到“反码”，是因为它太容易从电路上来实现了（仅需要取反即可）。

与前面介绍的“补码”一样，其实“反码”也可以说是专门针对负数的，因为正数的反码与补码、原码一样。它的编码规则为：对负机器数除符号位外的其他各位逐位取反，即原来为 1 就变为 0，原来为 0 就变为 1。由此可见，补码编码方式中也用到了“反码”。

【说明】其实反码在 IPv4 地址的子网掩码计算时是必须要用到的，因为我们知道，子网掩码是对 IPv4 地址中的网络 ID 部分保持不变，而主机 ID 部分则必须全为 0，这里就可能需要对 IPv4 地址中的网络 ID 部分的位进行取反，由 1 变为 0 了。有关 IPv4 地址方面将在第 9 章中进行介绍。

对于负机器数来说，反码只是对原码的除符号位外的其他各位取反（正机器数的反码与它的原码、补码都一样），所以它同样具有原码的一些不足：

（1）运算结果有时不正确。

如“2-3”采用反码运算的格式如下，结果是正确的：

$(0\ 0000010)_{反}+(1\ 1111100)_{反}=(1\ 1111110)_{反}=-1$。

但是在1-1的反码运算中，结果就不正确了：

$(0\ 0000001)_{反}+(1\ 1111110)_{反}=(1\ 1111111)_{反}=-0$，本来应该是+0的，结果就成了-0。

再如十进制数“123-121”，用反码加法运算就得到：

$(0\ 1111011)_{反}+(1\ 0000110)_{反}=(0\ 0000001)_{反}=1$，显然也不正确。

【注意】反码是相互的，如10011001的反码为11100110（符号位是不变的），相反11100110的反码也是10011001。另外，与原码一样，在反码表示形式中0也有“+0”和“-0”之分，对应的反码分别为0 0000000和1 1111111。

（2）0也有二义性。

+0的反码当然就是它的原码00000000（假设字长为8），-0的原码为10000000，由此可得出它的反码为11111111。即+0与-0的反码不一样。

综上可以得出：正数的原码、反码和补码都是一样的，而负数的这3种表示形式就不一样了，负数的反码是对原码中除符号位外的其他各位取反，而负数的补码是再对其反码加1，也就是先对其原码中除符号位的其他各位取反，然后再在最低位加1。

1.4.4 阶码

“阶码”主要用于本章后面将要介绍的浮点机器数的表示。在计算机存储器中表示一个浮点数时需要给出指数，而不是直接在存储器中存储这个浮点数，因为浮点数中的小数点位置是不确定的。而用来表示具体浮点数中小数点位置的这个指数就是这里所说的“阶码”，是用整数形式表示的。

对于任意一个二进制机器数N，均可用$N=S\times 2^P$的形式表示，其中S为尾数，P为阶码，2为阶码的底。P、S都用二进制数表示，S表示N的全部有效数字，P指明小数点的位置。这类似于十进制数中的科学记数法。当阶码为固定值时，表示的数称为“定点数”，这时阶码的意义不大；当阶码为可变时，表示的数称为“浮点数”，这是阶码的主要应用。

有关“定点数”和“浮点数”将在本章后面具体介绍。

1.4.5 移码

“移码”可以理解为“偏移的编码”，它有两个方面的应用：一是方便机器数的大小比较，二是用于浮点数中“阶码”的修正，它们都是通过在原来编码的基础上加上一个常数来实现的（但实现方式不一样）。下面具体就这两方面的应用进行介绍。

1. 在机器数比较方面的应用

我们已经知道，机器数的运算都是采用补码方式，因为采用补码运算时符号位与真值位的运算方法一样。但在进行机器数比较时，当机器数的真值用补码表示时，由于符号位和数值部分一起编码，与习惯上的表示法不同，因此人们很难从补码的形式上直接判断其真值的大小。

如十进制数 X=31 对应的二进制数为+11111，则$[X]_{补}$=011111；十进制数 X=-31，对应的二进制数为-11111，则$[X]_{补}$=100001。看上去好像 100001>011111，其实正好相反。如果我们对每个机器数的原码加上一个 2^n（n 代表原来二进制编码的真值部分位数，原码符号位用“+”或“-”表示），则 31 对应的+11111，再加上 2^5，即得到 11111+100000=111111；而-31 对应-11111+2^5（此时相当于 100000-11111），即得到 000001。然后拿两数转换后得到的结果 111111 和 000001 就可以直接通过二进制代码比较大小了。

移码的计算公式：$X_{移}=X+2^n$。X 代表原机器数的原码（符号位用“+”或“-”表示），n 代表原来二进制编码的真值位数。如果要计算 11010101 这个机器数的移码，首先去掉最高的符号位得出真值一共有 7 位，由此可以得到该数的原码为-1010101，再加上 2^7，即-1010101+1000000=00101011，这就是 11010101 这个机器数的移码。

其实以上公式也是移码与阶码的转换关系，即 $X_{移}=X_{阶}+2^n$，因为阶码通常是原码形式。由此可以得出移码与补码的关系：移码是补码的符号位取反，其他位不变。这样一来，在移码中，正数的符号位为 1，负数的符号位为 0，与前面的原码、反码和补码中的符号位规定相反，即补码与移码的关系是符号位互反的关系。

我们再用移码与补码的以上这种关系来计算前面 11010101 这个机器数的移码。此时要先计算出 11010101 这个机器数的补码。补码是除符号位外其他各位取反，然后再加 1 得到。11010101 的除符号位外其他各位取反后的值为 10101010，再加 1 就得到 10101011。再把最高的符号位取反（其他位不变），得到 11010101 的移码为 00101011，与前面通过移码公式计算得出的结果完全一样。

【说明】有关机器数的算术运算方法将在本章后面介绍。

2. 在浮点数阶码中的应用

前面说到了浮点数的“阶码”用来表示指数大小，也有正负之分。但这样一来，一个机器数在存储器中存储时就可能要包括两个符号位：一个为机器数本身的符号位，一个为阶码的符号位。

为了节省存储空间，省去阶码的符号位，就采用了对每个阶码都加上一个相同的、比较大的正常数（称偏移值），这样就使所有阶码值都为正整数（可以省去符号位）。并且在运算时系统会自动对指数部分减去某个固定的常数，就可以保持运算结果不变。这时用来表示浮点数小数点位置的“阶码”就变成“偏移”了的阶码，也就是这里所介绍的“移码”，又称“增码”。

在浮点数的阶码中所加上的这个常数称为“偏移值”，具体将在本章后面介绍。

1.5 机器数的分类

前面说到，机器数就是计算机中参与运算的有符号二进制数。其实它与我们平时使用的十进制一样，有整数也有小数。但关键的不同点是在存储器中并没有为小数点提供二进制码位，这就决定了机器数在存储器中的表示形式与我们平时书写十进制数时不太一样，需要对小点数进行特殊处理，这也就有了下面将要介绍的“定点数”与“浮点数”之分。

1.5.1 定点数

整数类型的机器数很好表示，只是在最高位多了一个符号位，小数点完全可以忽略，就像我们在书写十进制整数时一样。但如果是小数，小数点是不能不写的，在计算机中也不是像我们平常书写时那样好处理了，因为计算机存储器是没有专门的位来表示小数点的。

为了能使计算机自动识别机器数的小数点位置，正确识别对应机器数的大小，依据机器数中小数点的位置是否固定把整个机器数分为“定点数”（Fixed Point Number）和“浮点数”（Floating Point Number）两大类。本节先介绍定点数，下节再介绍浮点数。

从名称上很容易知道，定点数是小数点固定在某一个位上的机器数。它又可分为“定点整数”和“定点小数”，就像十进制数中包括了纯整数和纯小数一样，当然在一个机器数中也可以同时包括整数和小数。

1. 定点整数

“定点整数”是小数点固定在有效数值部分最低位之后的定点数，是纯整数。如一个定点整数是 n 位，即 $X=X_0X_1X_2\ldots X_{n-1}$，最高位 X_0 为符号位，则它所能表示的数值范围为 $-(2^{n-1}-1)\leqslant X\leqslant 2^{n-1}-1$。例如一个 8 位的定点整数，除了符号位外，用于表示真值的共 7 位，由此可以得出它能表示的最大值是 01111111，即 127，等于 2^7-1，所能表示的最小值为 11111111，即-127，等于 $-(2^7-1)$。

2. 定点小数

“定点小数”是小数点位置固定在符号位之后、有效数最高位之前的定点数，是纯小数。如一个定点小数是 n 位，即 $X=X_0X_1X_2\ldots X_{n-1}$，最高位 X_0 同样为符号位，则它所能表示的数值范围为 $-(1-2^{-(n-1)})\leqslant X\leqslant 1-2^{-(n-1)}$。例如一个 8 位的定点小数，除了符号位外，用于表示真值的共 7 位，由此可以得出它能表示的最大值是 0.1111111，即 0.5+0.25+0.125+0.0625+0.03125+0.015625+0.0078125=0.9921875，等于 $1-2^{-7}$，所能表示的最小值为 1.1111111，即-0.9921875，等于 $-(1-2^{-7})$。



经验之谈

根据以上给出的定点整数和定点小数所能表示的数值范围不等式可以得出以下规律：

- 一个 n 位连续为 1 的正二进制整数（即 n 位正二进制整数所表示的最大数）的值是 2^n-1。例如在 IPv4 地址中，每个字节是 8 位，它的最大值（即每位为 1）就是 $2^8-1=255$。同理，一个 n 位连续为 1 的负二进制整数（即 n 位负二进制整数所表示的最小数）的值是 $-(2^n-1)$。
- 一个 n 位连续为 1 的正二进制小数（即 n 位正二进制小数所表示的最大数）的值是 $1-2^{-n}$，也就是不要把每位转换成十进制再相加了，只需要用 1 来减 2 的-n 次方（此时只需要计算一个二进制小数位的值）。同理，一个 n 位连续为 1 的负二进制小数（即 n 位负二进制小数所表示的最小数）的值是 $-(1-2^{-n})$。

3. 同时包括整数和小数部分的定点数

以上的“定点整数”是纯整数，“定点小数”是纯小数，它们的小数点很好判定，运算起来也很方便。而在实际运算中一般不太可能全是纯整数和纯小数的数，也就是大多数情况下是同时包括了整数和小数的定点数。

对于同时包括了整数和小数的定点数，为了能确定其小数点的位置，在一个字长的存储位中小数点位置必须事先确定下来。这也就固定了一个字长所能表示的机器数的大小精度了。如一个 8 位字长，除了符号位还有 7 位用来表示真值，如果规定小数点是在第 3 位，则用于表示机器数真值的 7 位中，最高 4 位用来表示整数，最低 3 位用来表示小数。这样一来，真值整数部分最大值的绝对值部分就是 2^4-1=15，真值的精度就是保留 3 位小数。

很显然，定点数表示法的缺点在于其形式过于僵硬，固定的小数点位置决定了固定位数的整数部分和小数部分，不利于同时表达特别大的数或特别小的数，所以在实际的计算机运算中是很少采用这种格式的，而是采用我们下面将要介绍的“浮点数”。

1.5.2 浮点数

“浮点数”是小数点位置可变的机器数，采用了类似十进制数中的科学记数（Scientific Notation）法的表达方式。如十进制数中，123.45 用科学记数法可以表达为 1.2345×10^2，其中 1.2345 为尾数，10 为基数，2 为指数，尾数和指数均为十进制数。如果是负数的话，前面还有一个负号（-）。

如果把十进制数用 N 表示，尾数用 M 表示，指数用 E 表示，十进制数的科学记数法的表达公式就是：$N=\pm M\times10^E$。

按照这个思路，把以上的十进制数换成二进制机器数，则可得出二进制数的类似科学记数法表达公式为：$N=\pm M\times2^E$。其中，基数由原来的 10 换成 2，尾数 M 只能是二进制数（原码格式），指数 E 是对应十进制数的二进制数（也是原码格式，称为“阶码”，在下面将要介绍的 IEEE 754 浮点数中也可用“移码”表示）。浮点数在计算机存储器中存储的格式可以是原码或补码格式，通常是采用补码格式，因为补码格式对于负数也是无异义的。

以上的浮点数表示形式也就决定了浮点数在存储器中的存储格式，因为它的小数点位置是可变的，所以肯定不能直接把浮点数的二进制表示式在存储器中存储，否则就无法知道这个数到底多大了。

从前面得出的浮点数表示形式可以看出，一个浮点数只要知道它的符号位、指数和尾数，就可以最终根据以上公式得出它的实际大小。所以浮点数在存储器中就是把以上包括 1 个符号位（用 S 表示）、指数 E 和尾数 M 这三个部分进行存储，如图 1-13 所示。当然关于 E 和 M 具体各占多少位要视不同的浮点数定义，下节将介绍 IEEE 754 标准下的浮点数定义。

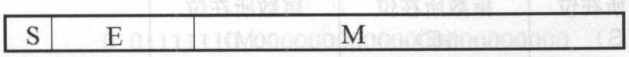


图 1-13 浮点机器数的存储格式

E 的二进制表示形式为定点整数，表示浮点数的小数点位置，决定了浮点数的表示范围；M 为定点小数，表示浮点数的有效值，决定了浮点数的精度。指数 E 为正时，表示转换后尾数部分

的小数点与原数的小数点位置相比左移了 E 位（二进制格式，计算位数时要转换为十进制），指数为负时，表示转换后尾数部分的小数点与原数的小数点位置相比右移了 E 位。

如一个浮点数为-0.10101×2^{11}，则它在 16 位字长的存储器（假设规定 E 用 4 位表示，M 用 11 位表示，M 符号位占 1 位）中的存储格式如图 1-14 所示。

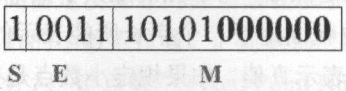


图 1-14　浮点数在存储器中的存储结构示例

由以上的分析可知，浮点数是一个纯小数和一个指数的乘积。但我们知道在十进制数的科学记数法中，同样的数值可以有多种浮点数表达方式，比如上面例子中的 123.45 可以表达为 12.345×10、1.2345×10^2 或 0.12345×10^3。在浮点数表示中也一样，尾数中的小数点位置也是可随意的。

为了规范起见，在机器数的浮点数表达形式中规定尾数的最高有效位（即小数点右边第一位）不能为 0，必须为 1。把不满足这一要求的尾数变成满足要求的尾数的操作过程称为浮点数的规格化处理，可通过尾数小数点的移位和修改“阶码”来实现。

1.5.3　IEEE 754 浮点数的分类

浮点数的定义有许多，目前通常是采用 IEEE 754 国际标准中的定义，把浮点数分成了单精度（Single）浮点数和双精度（Double）浮点数两种。在这个标准中规定，单精度浮点数长度为 32 位（4 字节），双精度浮点数长度为 64 位（8 字节）。

IEEE 754 标准中的以上两种浮点数的表示方式与前面介绍的浮点数表示方式在结构上是一样的，只不过在 IEEE 754 中把指数部分用“移码”来表示，而不是用以前的“阶码”，而且 E 和 M 的表示形式一般是要求用补码，而不是用原码，因为补码的符号位都参与运算，计算结果都没有错误。

前面我们知道，浮点数的指数 E 用阶码表示时是可正、可负的，这样就会使得一个浮点数中出现两个符号位：浮点数自身的和浮点数指数部分的。这样的结果是，在比较两个浮点数大小时显然比较麻烦。“移码”的定义在 1.4.5 节中介绍了，是在阶码基础上加一个固定的正常数得到，这个固定的正常数在 IEEE 754 中称为“偏移”。之所以要用移码表示，目的就是为了使指数部分均为无符号的正整数，这样补码与原码是一样的，不用转换。

在 IEEE 754 中，浮点数在存储器中存储的格式以及所指定的偏移值如表 1-4 所示。

表 1-4　单精度浮点数和双精度浮点数各部分在存储器中的结构

浮点数类型	符号所在位（S）	指数所在位（E）	尾数所在位（M）	偏移值
单精度浮点数	31(1)	30-23(8)	22-0(23)	127（对应二进制为：01111111）
双精度浮点数	63(1)	62-52(11)	51-0(52)	1023（对应二进制为：0111111111）

此时表中的 E 是以“移码”格式表示的，在 IEEE 754 中用于表示浮点数中小数点位置的“移码”=阶码+偏移值。如果用 e 来表示原来的阶码，则移码与阶码的关系为：

E=e+偏移值

在计算浮点数时可用以下公式统一表示：$X=S(1.M)\times 2^{E-偏移}$，X 为浮点数原码，S 为符号位，M 为尾数，E 为移码，“E-偏移”就等于阶码。

1.5.4 IEEE 754 浮点数的计算

给定一个机器数，要计算出其 IEEE 754 标准浮点数格式，可直接按上节介绍的公式：$S(1.M)\times 2^{E-偏移}$进行，关键的计算步骤如下：

（1）把要转换的数转换成二进制。

十进制数转换成二进制数要区分整数部分和小数部分，因为它们的转换方法是不一样的，具体转换方法参见 1.2.2 节。

（2）把二进制数转换成 $1.xxxx\times 2^e$ 格式。

这里的 e 就是“阶码”（此时用十进制数表示）。当 e 大于 0 时，表示转换成 $1.xxxx\times 2^e$ 格式后小数点相对原来的小数点位置左移了 e 位；当 e 小于 0 时，表示转换成 $1.xxxx\times 2^e$ 格式后小数点相对原来的小数点位置右移了 e 位。

（3）由原来数的正负属性得出 S 是 0（正数时）还是 1（负数时），并由上步得出的 e 值计算出 E（E=e+偏移），E 为移码。

（4）在 1.xxxx 的小数点右边第一位起插入 E 的二进制形式，同时补足尾数（即 xxxx 部分）对应精度位数，即得出浮点数的二进制表示形式。

【示例 1】求十进制数 0.5 的单精度浮点数。

（1）按照 1.2.2 节介绍的方法计算出十进制数 0.5 的二进制表示形式：0.1。

（2）转换为 $1.xxxx\times 2^e$ 格式：1.0×2^{-1}，即小数点相对原来右移了 1 位，阶码 e 为-1。

（3）根据“E= e+偏移”（E 为移码）得出 E=-1+127=126（单精度的偏移值为 127），转换成二进制后为 01111110。又因为原数是正数，所以 S=0。

（4）在以上 1.0 的小数点右边第一位后面插入 E，然后补足尾数对应精度位数（原来的尾数只有一位：0，单精度的尾数为 23 位）。

最终得到十进制数 0.5 所对应的单精度浮点数为：0 01111110 00000000000000000000000，具体格式如图 1-15 所示。

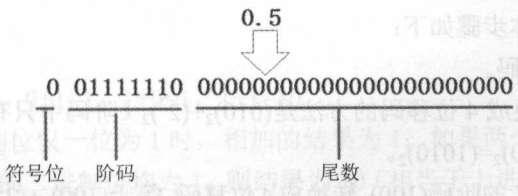


图 1-15 十进制数 0.5 的单精度浮点数格式

【示例 2】求十进制数-1.75 的单精度浮点数。

（1）按照 1.2.2 节介绍的方法计算出十进制数 1.75 的二进制表示形式：1.11。

（2）转换为 $1.xxxx\times2^e$ 格式：1.11×2^0，故得出 e 为 0，因为小数点没有移位。

（3）根据“E=e+偏移”得出 E=0+127=127，转换成二进制后为 01111111。单精度的偏移值为 127。又因为原数是负数，所以 S=1。

（4）在以上 1.11 的小数点后面第一位起插入 E，然后补足尾数对应精度位数（原来的尾数只有两位：11，单精度的尾数为 23 位），即需要在后面加 21 个 0。

最终得到十进制数-1.75 所对应的单精度浮点数为：1 01111111 11000000000000000000000，具体格式如图 1-16 所示。但此时尾数是原码格式，又因为尾数是负数（移码总是正数，其补码与原码格式一样），还需要转换成补码格式，最终得到该数的补码形式单精度格式为：1 01111111 01000000000000000000000。

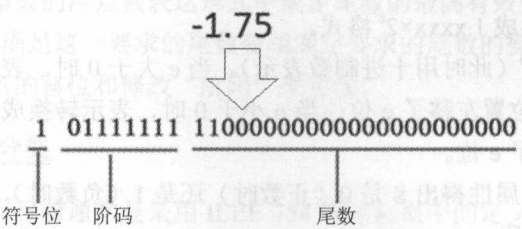


图 1-16　十进制数-1.75 的单精度浮点数格式

1.5.5　浮点数表示形式的转换

通过前面的学习已经知道，浮点数的表示形式还不是固定的，指数既可以用阶码表示，又可以用移码表示，尾数可以是原码形式，也可以是补码形式。本节就来举例说明这些不同表示形式之间的转换方法。

【示例】如有两个浮点数 $X=2^{010}\times0.11011011$，$Y=2^{100}\times(-0.10101100)$，当前指数采用的是二进制阶码格式，尾数采用原码格式。如果要把这两个浮点数转换成指数用 4 位移码，尾数用 9 位（含符号位）补码形式表示，该如何转换呢？

这样的题就有三项工作要做：一是把阶码转换成移码，这个在 1.4.5 节有介绍，并且给出了一个公式，即 $X_{移}=X_{阶}+2^n$（n 为原阶码的真值位数）；二是把原码形式的尾数转换成补码形式；三是确定浮点数的符号位。具体步骤如下：

（1）把阶码转换成移码。

把 X 的阶码$(010)_2$转换成 4 位移码的方法是$(010)_2+(2^2)_2$（阶码中只有 2 位是真值，也要转换成二进制形式）$=(010)_2+(1000)_2=(1010)_2$。

用同样的方法可以把 Y 的阶码$(100)_2$转换成 4 位移码，等于$(100)_2+(2^2)_2=(100)_2+(1000)_2=(1100)_2$。

（2）求尾数的补码形式。

先要得到尾数的原码形式。从两数的表示形式可以得出X的尾数（M）为11011011，Y的尾数（M）为10101100，且题中已说明已是原码形式。

因为X为正数，所以符号位（S）为0，而Y为负数，所以符号位（S）为1。通过前面的学习已经知道，浮点数的格式为：S E M，S为符号位，E为阶码或移码，M为尾数。此时如果浮点数中指数用移码表示，尾数M用原码表示，则X和Y的格式分别为：$X_{浮}$=0 1010 11011011，$Y_{浮}$=1 1100 10101100。

此时如果要把尾数M用9位补码表示，则需要将X和Y尾数部分转换成补码形式。

因为X是正数，所以它的尾数M的补码与原码一样，等于11011011（加上符号位正好为9位）。Y是负数，它的尾数M原码转换成补码时先要对除符号位外的其他各位取反（取反后得出低8位为01010011），然后再加1，得到01010100（加上符号位也正好为9位）。

（3）得出最终的补码表示形式。

通过以上步骤以及浮点数的符号位可以最终得出尾数M用补码形式表示的X、Y浮点数的存储格式为（正数X的补码格式与原码格式是完全一样的）：$X_{浮}$=0 1010 11011011，$Y_{浮}$=1 1100 01010100。

1.6 二进制数的运算

本节所介绍的二进数运算主要包括无符号二进制数四则算术运算、机器数补码形式的加/减法运算和无符号二进制数的逻辑运算。比较复杂的浮点数运算将在本章后面介绍。其实无符号二进制数的算术和逻辑运算方法同样适用于补码形式二进制数、浮点数中真值部分二进制数的对应运算。

【说明】通过前面的学习我们已经知道，因为补码形式的二进制数符号位也是直接参与运算的，不需要单独处理，所以在计算机中都是以补码形式来进行二进制数运算的。也正因为如此，在进行各种二进制运算时都建议把对应的二进制数转换为补码形式。

1.6.1 二进制数的算术运算

无符号二进制数的加、减、乘、除四则算术运算法则其实与十进制数的四则算术运算法则是一一对应的。你理解了十进制数的四则算术运算法则，无符号二进制数的四则算术运算就一点都不难了。

1. 加法运算

加法运算法则：0+0=0，0+1=1，1+0=1，1+1=10。

即当两个相加的二进制位仅一位为1时，相加的结果为1；如果两个二进制位全是0，相加的结果仍为0；如果两个相加的二进制位均为1，则结果为10（相当于十进制中的2），要向高位进1，也就是“逢2进1”规则，与十进制中的“逢10进1”道理一样。

【说明】在进行二进制加减法运算时，最关键的一点就是逢 2 进 1，进 1 当 1，而借 1 当 2。大家联想一下我们经常使用的十进制数加法运算法则，那就是逢 10 进 1，进 1 当 1，而借 1 当 10，这样一来我们就好理解了，无符号二进制数的加法运算法则只是把原来十进制数加法运算法则中的 10 改成了 2。

【示例 1】计算$(10010)_2$+$(11010)_2$的过程如图 1-17 所示（注意两数要从最低位开始对齐）。

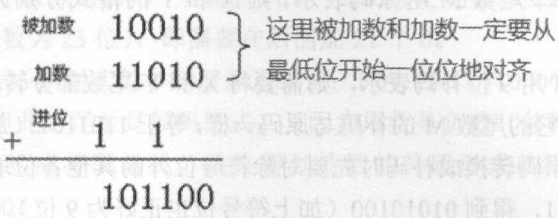


图 1-17　二进制加法运算示例

（1）进行最低位相加，这里加数和被加数都为“0”，根据加法原则可以知道，相加后为“0”。

（2）进行倒数第二位相加，这里加数和被加数都为“1”，根据加法原则可以知道，相加后为“$(10)_2$”，此时把后面的“0”留下，而把第一位的“1”向高一位进“1”。

（3）进行倒数第三位相加，这里加数和被加数都为“0”，根据加法原则可以知道，本来结果应为“0”，但倒数第二位已向这位进“1”了，此时就要同时把“被加数”“加数”和“进位”这三个数加起来，所以结果应为 0+0+1=1。

（4）进行倒数第四位相加，这里加数和被加数分别为“1”和“0”，倒数第三位也没有进位，根据加法原则可以知道，相加后为“1”。

（5）最高位相加，这里加数和被加数都为“1”，根据加法原则可以知道，相加后为“$(10)_2$”。同样需要把第一位的“0”留下并向高位进 1，这样会产生新的最高位，值为“1”（但如果超出了字长的限制，则新产生的最高位将溢出）。

这样$(10010)_2$+$(11010)_2$的最后运算结果为 101100。

2. 减法运算法则

减法运算法则：1-1=0，1-0=1，0-0=0，0-1=-1。

即当两个相减的二进制位中同为 0 或 1 时，相减的结果为 0；如果被减数的二进制位为 1，而减数的二进制位为 0，则相减的结果仍为 1；如果被减数的二进制位为 0，而减数的二进制位为 1，则需要向高位借 1，但此时是借 1 当 2，与十进制中的借 1 当 10 道理一样。

【示例 2】计算$(111010)_2$-$(101011)_2$的过程如图 1-18 所示（注意两数要从最低位开始对齐）。

（1）最低位相减，这里被减数为“0”，减数为“1”，不能直接相减，需要向高位（此时为倒数第二位）借“1”，这样相当于得到了十进制中的“2”，用 2 减 1 结果就得到 1。

（2）对倒数第二位相减，此时本来被减数和减数均为“1”，但是被减数的该位被上一步借走了 1，所以最后就变为“0”（1-1）了。此时也不能直接与减数相减了，又需要向高位（此时为倒

数第三位）借 1。同样是借 1 当 2，相当于该位总共为 0+2=2。这样倒数第二位相减后的结果为 2-1=1。

```
被减数  111010   ┐
减数    101011   ┘ 这里被减数和减数一定要从最低位开始一位位地对齐
借位     1111
-------------------
        001111
```

图 1-18　二进制减法运算示例

（3）用上一步同样的方法计算倒数第三位和倒数第四位的减法运算，结果都为 1。

（4）计算倒数第五位的减法运算，此时被减数原来为“1”，可是已被倒数第四位借走了 1，所以成了“0”（1-1），而此时减数也为“0”，可以直接相减，得到的结果为“0”。

（5）计算最高位的相减，被减数和减数均为“1”，可以直接相减，得到的结果为“0”。

这样一来，$(111010)_2$-$(101011)_2$ 的结果是$(001111)_2$，由于整数的前导 0 可以不写，所以最后结果就是$(1111)_2$。

3. 乘法运算法则

乘法运算法则：0×0=0，0×1=0，1×0=0，1×1=1。

即只有当两个相乘的二进制数都为 1 时，相乘的结果才为 1；两个相乘的二进制数中只要有一位为 0（也包括是两位同时为 0），则相乘的结果都为 0。也可以这么理解：1 与任何数相乘的结果就是对应的被乘数；而 0 与任何数相乘的结果都为 0。这与十进制中的乘法运算法则也是一样的。

【说明】在乘法运算中，乘数的每一位要与被乘数的每一位分别相乘，而不仅是对应位相乘，而且每一位乘数与被乘数的每一位相乘的结果的最低位要与对应的被乘数位一样。当然这与十进制的乘法运算法则也是一样的，很好理解。

【示例 3】计算$(1010)_2$×$(101)_2$ 的过程如图 1-19 所示（注意两数要从最低位开始对齐）。

```
被乘数  1010
×  乘数  101
-------------------
        1010   ┐
        0000   ├ 这里每行的最低位一定要与对应的乘数位对齐
+ 1010         ┘
-------------------
积  110010
```

图 1-19　二进制乘法运算示例

（1）乘数的最低位与被乘数的所有位相乘，因为乘数的最低位为“1”，根据乘法原则可以得

出，结果其实就是被乘数本身，直接复制下来即可。此处结果就为 1010。

（2）乘数的倒数第二位与被乘数的所有位相乘，因为乘数的这一位为“0”，根据乘法运算法则可以得出，结果均为“0”。此处结果就为 0000。

（3）乘数的最高位与被乘数的所有位相乘，此时乘数为“1”，结果就是被乘数本身。此处结果就为 1010。

（4）按照前面介绍的二进制加法原则对以上三步所得的结果按位相加（但这里的位对齐方式与单纯的二进制数相加不一样，最低位一定要与对应被乘数位一致。这也与十进制的乘法运算方法一样），结果得到$(110010)_2$。

4. 除法运算法则

当被除数大于除数时，商是“1”；当被除数小于除数时，不够除，商只能是“0”，这与十进制的除法也类似。二进制只有两个数（0 和 1），因此它的商也只能是 1 或 0。

【示例 4】计算$(11001)_2 \div (101)_2$的过程如图 1-20 所示。

```
        商   101
除数  101 ) 11001  被除数
          - 101
            101
          -  101
          -------
             000
```

图 1-20　二进制除法运算示例

（1）因为除数为“101”，有 3 位，所以在被除数中也至少要取 3 位（从最高位开始取）。被除数的高 3 位为“110”，恰好比除数“101”大，可以直接相除，但商只能是 1（因为二进制的最大数元就是 1），然后把被除数与用商“1”与除数相乘后的结果进行相减，得到的值为“1”。

（2）从被除数中取下一位“0”下来，与上一步的差“1”值组成新的被除数，为“10”，显然它比除数“101”小，不够除。于是在商的对应位置上输入“0”。

（3）继续从被除数中取下一位“1”下来，与上一步的余数“10”值组成新的被除数，为“101”，此数正好与除数“101”相等，所以此时的商取“1”正好除尽。

这样一来$(11001)_2 \div (101)_2$所得的商就是$(101)_2$。

1.6.2　补码的加/减法运算

通过前面的学习我们已经知道，机器数的补码可由原码和反码得到。如果机器数是正数，则该机器数的补码与原码、反码一样；如果机器数是负数，则该机器数的补码是对它的反码在末位加 1 而得到的。

1. 补码的加法运算

补码的加法运算法则：$[X+Y]_{补}=[X]_{补}+[Y]_{补}$。

该式表明，两个机器数相加的补码可以通过先分别对两个机器数求补码，然后相加得到。在采用补码形式表示时，进行加法运算可以把符号位和数值位一起进行运算（若符号位有进位，则溢出不管），结果为两数之和的补码形式。

【示例 1】如要求两个十进制数：35+18 的补码（假设字长为 8）。根据上面的补码加法运算法则可以得知，只需分别求 35 和 18 这两个数的补码，然后相加即可。又因为这两个数都是正数，所以它们的补码与原码一样。

这样一来，这道题实际上也就是求 35 和 18 这两个十进制数的原码和。35 的原码为 0 0100011（注意：最高位为符号位），18 的补码为 0 0010010。所以 35+18 的补码就等于(0 0100011)B+(0 0010010)B=(00110101)B，如图 1-21（a）所示。如果转换成十进制则等于 53，结果正确。如果相加后有超过字长的位溢出，则直接丢弃。

【示例 2】如果要求两个十进制数：35+(-18)和的补码也是直接求正数 35 和负数(-18)的补码和。正数 35 的补码与其原码一样，前面已计算出，为 0 0100011；而后面那个“-18”因为是负数，不能直接从它的原码得到补码了。需要先就要求（-18）的原码（1 0010010），然后对其除符号位外的其他各位取反，得到其反码（为 1 1101101），最后再在其末位（最低位）加 1，最终得到其补码为 1 1101110。

这样一来，“35-18”的补码就是(00100011)B+(11101110)B，结果为 00010001，如图 1-21（b）所示。这里要注意，两个补码相加后产生了第 9 位（为 1）的溢出，直接丢弃，所以结果就是(00010001)B。如果转换成十进制则等于 17，结果正确。

```
    0 0100011          0 0100011
+   0 0010010      +   1 1101110
-------------      -------------
    0 0110101      1 0 0010001
                   ↑
                   新产生的最
                   高位，溢出
```

（a）　　（b）

图 1-21　两个补码加法运算示例

2. 补码的减法运算

补码的减法运算法则：$[X-Y]_{补}=[X]_{补}+[-Y]_{补}$。

该公式表明，求两个机器数的差值（如$[X-Y]_{补}$）的补码可以通过求被减数的补码（如$[X]_{补}$）与减数的负值的补码（$[-Y]_{补}$）的和得到。

负数补码的求解方法我们在前面已有介绍，那就是先对除符位号外的其他各位取反，最后加 1。其实还有一个更简单的方法，那就是负数的补码等于该正数的补码（也等于该正数的原码）全部位（含符号位）取反，然后再加 1（因为正数与负数的符号也正好相反）。如已知$[15]_{补}=00001111$，则$[-15]_{补}=11110000+1=11110001$。

【示例 3】现假设 X=+35，Y=+18，要求$[X-Y]_{补}$（假设字长为 8）。

先根据正数的补码与原码一样的规则，求得$[X]_{补}$= 00100011，$[Y]_{补}$ = 00010010；再根据以上介绍的负数求补码操作规则（全部位取反，再加 1），可以得到$[-Y]_{补}$=11101110。

最后用$[X]_{补}$+$[-Y]_{补}$公式即可得到最终的$[X-Y]_{补}$ = 00010001，如图 1-22（a）所示。转换成十进制也可得到结果为 17，正确，且与上面使用加法法则运算的结果一样。注意，这里相加的结果也产生了溢出的第 9 位（1），直接丢弃。

【示例 4】如有两个十进制数 X=-35，Y=-18，现要求$[X+Y]_{补}$的值。

根据前面介绍的加法法则得知，可先求得$[X]_{补}$和$[Y]_{补}$的值，然后再相加，即$[X]_{补}$+$[Y]_{补}$。根据本章前面介绍的求负数补码的方法来计算-35 和-18 的补码。

上一示例中已求出了 35 和 18 的补码（也就是它们的原码）分别为 00100011 和 00010010，然后分别对它们各位取反，再加 1，由此可得出$[-35]_{补}$为 1 1011101，$[-18]_{补}$=11101110。

最后$[-35]_{补}$+$[-18]_{补}$的运算过程如图 1-22（b）所示，结果为 11001011。注意这是一个有符号位数，所以结果为-75。这样看起来结果是不正确的，因为-35-18 应该等于-53。这时就要特别注意了，这个-75 是一个补码形式，要看最终的结果还得把它转换成原码。按照上面介绍的方法立即可以得到$[-75]_{补}$的原码为 10110101（注意它也是一个有符号位数，最高位的 1 代表为负数）=$[-53]_{原}$。这样结果就正确了。

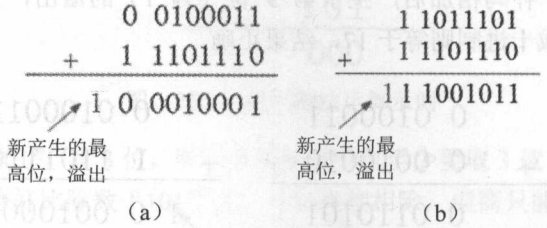


图 1-22　两个补码减法运算示例

1.6.3　二进制数的逻辑运算

逻辑运算是指对因果关系进行分析的一种运算，这也是在计算机中经常采用的一种二进制运算。如在通过 IPv4 地址和子网掩码得出其网络地址时就要用到逻辑“与”运算（具体将在本书后面相应章节介绍）。逻辑运算的结果并不表示数值大小，而是表示一种逻辑概念，若成立则为“真”，或用“1”表示；若不成立，则为“假”，或用“0”表示。

二进制数的逻辑运算主要有“与”“或”“非”和“异或”4 种。

1. “与”运算（AND）

“与”运算又称逻辑乘，用符号“.”或“∧”来表示。运算法则如下：

0∧0 = 0　　0∧1 = 0　　1∧0 = 0　　1∧1 = 1

即只要两个参加“与”运算的数的对应位的数中有一个为 0，则运算结果均为 0；仅当两数的

对应位均为 1 时结果才为 1，很容易判断。这与前面介绍的二进制乘法运算是一样的。

【示例 1】图 1-23 所示是两个“与”的逻辑运算示例。左图所示的是两个位数不一样的二进制数进行“与”运算，这时要求两个数从最低位开始对齐，在位数少的二进制数的最高位前面加上“0”补齐，使得它与位数多的二进制数有一样的位数。

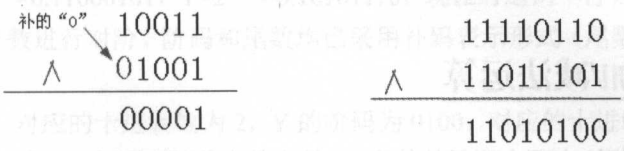


图 1-23 两个“与”逻辑运算示例

2. “或”运算（OR）

“或”运算又称逻辑加，用符号“+”或“∨”表示。运算法则如下：

$0 \vee 0 = 0 \quad 0 \vee 1 = 1 \quad 1 \vee 0 = 1 \quad 1 \vee 1 = 1$

即只要两个参加“或”运算的数的对应位的数中有一个为 1，则运算结果为 1，仅当两数的对应位均为 0 时结果才为 0，也很容易判断。

【示例 2】如图 1-24 所示是两个“或”逻辑运算的示例。同样，进行“或”运算时要求两数从最低位开始对齐，位数少的数在最高位前面加“0”补齐，最终使两个二进制数的位数相同。

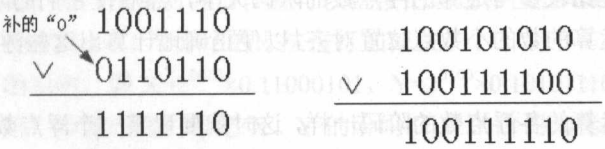


图 1-24 两个“或”逻辑运算示例

3. “非”运算（NOT）

“非”运算就是逐位求反的运算，其运算法则为：“0”的反值为“1”，“1”的反值为“0”，也就是“0”与“1”互为反。注意：“非运算”只是针对一个二进制数进行的，这与前面的“与”和“或”运算不一样。如“101110101”进行“非”运算后就得到“010001010”（可简写为“10001010”）。

4. “异或”运算（XOR）

“异或”运算用符号“⊕”来表示。运算法则如下：

$0 \oplus 0 = 0 \quad 0 \oplus 1 = 1 \quad 1 \oplus 0 = 1 \quad 1 \oplus 1 = 0$

即当两个参加“异或”运算的二进制数对应位相同时运算结果为 0，不同时运算结果为 1。

【示例 3】图 1-25 所示是两个“异或“逻辑运算的示例。同样，进行“异或”运算时要求两数从最低位开始对齐，位数少的数在最高位前面加“0”补齐，最终使两个二进制数的位数相同。

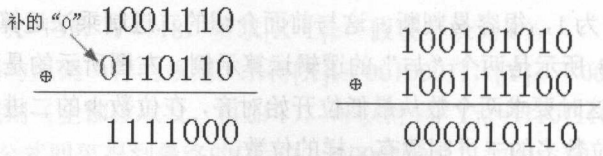


图 1-25　两个“异或”逻辑运算示例

1.7　浮点数的加/减法运算

浮点数与定点数相比有两个明显的特点：一是小数点位置不固定，而我们知道在做加减运算时，两数的小数点位置必须对齐；二是在存储器中不是直接把浮点数本身直接存储，而是在中间插入了一个用于表示小数点位置的阶码或移码，所以不能直接把在存储器中存储的浮点数相加减，只需对代表浮点数有效数部分的尾数进行加减运算。

正因为如此，浮点数的加减法运算相比定点数的运算就要复杂许多，当然从事网络职业的人士也没必要做太深的了解（从事程序开发的人士是必须要掌握的）。有关浮点数的加减法运算包括以下 5 个基本步骤：对阶、尾数运算、规格化处理、舍入处理、溢出处理。

1.7.1　对阶

所谓“对阶”就是比较参与运算的浮点数的阶码大小，然后使它们的阶码（或移码）都一样，其最终目的是使参与运算的数的小数点位置对齐，以便正确地计算出这些浮点数之间的加减法运算结果。

既然是要使参与运算的各浮点数的阶码一样，这时就要以某一个浮点数的阶码为标准了。通常是以大阶码为准，即阶码小的数要向阶码大的数对齐，即把小阶码的值调成与大阶码的值一样。这时为了确保小阶码的数最终值与原来保持不变，肯定要对小阶码的浮点数的尾数的小数点进行移位。就像十进制数中的 1.2345×10^3，现在把指数由原来的 3 改为 5，则原来的尾数 1.2345 中的小数点肯定不能还保持不变，要向左移两位（因为指数增加 2），变成 0.012345，最终的形式为 0.012345×10^5。

因为是以大阶码为准，相当于小阶码的数的阶码要增大，所以其尾数肯定要做相同倍数的减小（这样才能保持原数值不变），即阶码小的尾数要向右移（小数点是向左移）对应的位数。移位原则是：在尾数最高位前面加对应位数的 0（小了多少位就加多少个 0），然后原尾数最后的 n 位被丢弃（移了多少位就丢弃最后面多少位），因为用于存储尾数部分的存储器位数是固定的。



至于对阶中的尾数移位可以借助我们常用的十进制数科学记数法来帮助理解。假设十进制数 1234.5 原来表示成 1.2345×10^3，现在要表示成$\times10^6$的格式，则对应的格式为 0.0012345×10^6格式（在

原来的小数点右边最高位前面移了 3 位，补加了 2 个 0）。又假设小数点后面最多只能 4 位（就相当于浮点数中用来保存尾数的位置是固定的），这样一来，最终的存储格式就为 0.0012×10^6，原来最后面的“345”就丢弃了。当然这样会与原数在大小上有一些区别（因为最终成了 1200），但仍能最大限度地保持不变。

【示例】如 $X=2^{0010}\times0.11000101$，$Y=2^{0100}\times0.10101110$，现在对这两个浮点数进行加法或减法运算，首先就要对这两个数进行对阶。阶码和尾数均已采用补码表示形式（尾数的符号位体现在浮点数的符号位上）。

X 的阶码为 0010，对应的十进制数为 2，Y 的阶码为 0100，对应的十进制数为 4，由此可知 X 的阶码比 Y 的阶码小 2。此时把 X 的阶码也变为与 Y 的阶码一样，即都调成 4，同时把 X 的尾数小数点左移 2 位（在小数点右边最高位前面加两个 0，原来最低的 2 位被丢弃），这样才能使 X 的值与原来保持基本不变。

X 原来的尾数为 11000101（符号位体现在浮点数的符号位上，此数为正），向右移两位（仍要保持原来的总位数不变，则要在前面补相应位数的 0，原来最右边的对应两位要丢弃），则变成 00110001。这样一来，X 的阶码也变成 0100，最终 X 在存储器中的位结构变成了 0 0100 **00110001**（原来为 0 0010 **11000101**）。

1.7.2 尾数运算

参与运算的浮点数的小数点位置对齐后，就可以直接把经过移位的尾数进行相加或相减运算。这一步很简单，可以直接按照 1.6.1 节介绍的方法进行加减法。

【示例】继续前面的示例，即 $X=2^{0010}\times0.11000101$，$Y=2^{0100}\times0.10101110$，现要求 X+Y。

在上节介绍的“对阶”步骤中我们已把 X、Y 的阶码以 Y 的阶码为标准进行了统一，也求出了对阶后 X 的尾数为 00110001，而 Y 的尾数不变，为 10101110，把两者直接相加，得到的结果为 11011111，如图 1-26 所示。

$$\begin{array}{r} 00110001 \\ +\ 10101110 \\ \hline 11011111 \end{array}$$

图 1-26 X 和 Y 的尾数相加

1.7.3 规格化处理

进行加减法运算后的尾数可能不是规格化的数，就需要进行规格化处理。规格化的尾数格式要求如下：

- 尾数采用原码表示形式时：正数的规格化格式为：0.1xxxx，负数的规格化格式为：1.1xxxx。
- 尾数采用补码表示形式时：正数的规格化格式为：0.1xxxx，负数的规格化格式为：1.0xxxx。

以上最高位均代表符号位（0 代表正数，1 代表负数），关键是看小数点后面的数的格式。

- 对于双符号位（用两位来表示符号）的补码形式尾数，正数的规格化格式为 00.1xxxx，负数的规格化格式为：11.0xxxx，以上最高两位代表符号位（00 代表正数，11 代表负数），关键是看小数点后面的数的格式。

【说明】引入“双符号位”的设计的目的就是为了能快速检测出运算结果是否有溢出，因为双符号位规定了“00”代表正数，“11”代表负数，如果最高两位不是两种形式，而是“01”或者“10”就能快速地知道有溢出了。符号位为 01 时，称为上溢，即最高真值位相加后有进 1；为 10 时，称为下溢，即最高真值位相减后有借 1。在此我们仅介绍单符号位形式。

凡不符合以上格式要求的尾数均要进行规格化处理。对以上规格化格式进行总结可以得出：符号位与尾数最高位不一致才算是规格化，一致为非规格化。如 1.0xxxx、0.1xxxx 之类的尾数（最前面的为符号位）就是规格化的数，而 1.1xxxx，0.0xxxx 之类的数则是非规格化的数。另外，如果尾数大于 1，也是非规格化数。

对于非规格化的尾数需要进行相应的处理，这些处理方式又分“左规”和“右规”。所谓“左规”就是尾数要向左移位，每移 1 位阶码值减 1，直到为规格化数为止；对应“右规”就是尾数要向右移位，每移 1 位阶码值加 1，直到为规格化数为止。

具体什么情况下要采用左规，什么情况下要采用右规，基本原则如下：

- 运算结果产生溢出（由于原来两尾数的最高有效位相加有进位或者相减有借位时形成的）时，必须进行右规。如双符号位情形下的运算结果为 10.xxxx 或 01.xxxx 格式就是不符合规格化要求了，因为双符号位时符号位为“10”和“01”都是不正确的。右规时最高有效位前补相应位数的 0。此时 10.xxxx 格式右规后的格式为 11.0xxx，而 01.xxxx 格式右规后的格式为 00.1xxx。
- 如运算结果出现 0.0xxx 或 1.1xx（即符号位与尾数最高有效位相同）时，必须进行左规。左规时最低有效位后补相应位数的 0。

在前面的例子中已得出 X 和 Y 的尾数之和为 11011111，又因为它是正数，所以可表示成 0.11011111，已是规格化，不用再进行处理了。

假设某 X 与 Y 浮点数相减，并且得出的尾数之差为负数，如 1.11011111，这时就要进行规格化处理了，因为符号位 1 与尾数的最高位 1 相同。这时要对尾数左规，即向左移位。11011111 要向左移 2 位才可能使尾数的最高有效位与前面的符号位 1 不一样，即移位后的尾数是 01111100（最后两个 0 是补上去的）。同时阶码要从原值（0100）相应减 2（即减 0010），得到 0010。

1.7.4 舍入处理

在“对阶”和向左、向右规格化处理时，尾数要向左、向右移位。这样被移位的尾数与实际的尾数之间就可能存在一定误差，因此要进行舍入处理，以尽可能减小这种误差。

简单的舍入方法有以下两种：

（1）“0 舍 1 入”法。

类似于我们十进制数中的“四舍五入”法。即如果左规或右规时被丢弃的数位为 0，则舍去不计，反之要将尾数的末位加“1”。

如前面提到的 X 与 Y 的尾数和为 1.11011111（最高位为符号位），需要左规，尾数要向左移两位，得到 1.01111100，由此可见左移时去掉的是前面两位 1，需要在尾数的最后一位加 1。这样进行舍入处理后就得到 1.01111101。

（2）“恒置 1”法。

只要有数位被左规右规丢弃掉，就要在尾数的末位恒置“1”。此法精确度不高，因为从概率上来讲，丢弃的 0 和 1 各为 50%。

【说明】在 IEEE 754 标准中有更多的舍入模式，但比较复杂，在此不做介绍。

1.7.5 溢出处理

是否有溢出，在本章前面介绍了，如果采用双符号位补码形式，出现的双符号位为“01”或“10”时就会自动判定有溢出。但是如果采用的是单符号位，就不好判定了。这时要根据在对尾数进行规格化处理后“阶码”所能表示的取值范围是否超出了对应阶码位所能表示数的范围来定了。

前面已介绍到，右规后阶码要加相应的值，尾数每向右移一位加 1，最终可能造成新的阶码超出了原来用于表示阶码位数所能表示的最大数值（称为“上溢”）。

如某$[X+Y]_{补}$浮点数经过“对阶”和尾数相加后得到的尾数为 10.101110011，显然它需要右规，结果为 11.01110011，相当于尾数向右移了一位，原来尾数的最高有效位“1”被丢弃了，同时阶码也要相应加 1。再假设用来存储阶码的位数为 4（包括一位阶码符号位，所能表示的最大阶码数为+7），因原阶码为 0111，尾数右移一位后，阶码要加 1 就为 8 了，而这超出了 4 位阶码中的 3 个真值位所能表示的最大数+7，所以这时就要做溢出处理了。

当然，当尾数被左规时也可能造成溢出，因为左规后可能超出了用于存储阶码的存储位所能表示的最小值（称为“下溢”）。如左规后得到的新阶码为-5，假设阶码规定是用 4 位来存储的，现在要求尾数再向左移 3 位，这样阶码也要减小 3，得到-8，而-8 超出了 4 位阶码（最高位为阶码符号位）所能表示的最小数-7。

如果发现有溢出，计算机设备会根据以下原则进行处理：

- 如果是上溢，则停止运算，做中断处理。
- 如果是下溢，整个尾数按 0 处理。

1.8 信息编码

“数”不仅仅用来表示“量”，它还能作为代码（Code）来使用。例如，在我国实行的公民身份证制度，身份证上有一组 18 位的数为“身份证号码”。这就是用数字进行编码的例子。再如每一个学生入学后都会有一个学号，这也是一种编码。

编码的目的之一是为了便于标记特定的对象。为了便于记忆和查找，在设计编码时需要按照一

定的规则。这里介绍最常用的几种计算机编码，如 ASCII 码和一些汉字编码。

1.8.1 西文编码

西文编码就是对西文字母、符号或者操作进行的编码，常见的有 ASCII 码（American Standard Code for Information Interchange，美国标准信息交换代码）和 Unicode 码两种。

1. ASCII 码

ASCII 是基于拉丁字母的一套电脑编码系统，主要用于显示现代英语和其他西欧语言。它是现今最通用的单字节编码系统，并等同于国际标准 ISO/IEC 646。目前分为“标准 ASCII 码”和“扩展 ASCII 码”两种。

（1）标准 ASCII 码。

标准 ASCII 码采用 7 位二进制进行编码（共 8 位，但仅由低 7 位对信息进行编码，最高位为奇偶校验位），可组合表示 128（2^7）种状态，对应 128 个（对应十进制取值为 0～127）字符（或控制符）。其中包括 26 个英文大写字符、26 个英文小写字符、10 个数字字符、33 个标点符号和 33 个控制符。例如大写 A 的 ASCII 码是 65，小写 a 是 97。

【说明】奇偶校验是用来校验信息在传输过程中是否出现了差错的一种方法，分奇校验和偶校验两种。这两种校验方法分别是在信息传输前通过先在信息码的最高位前面加上校验码，使得最终所传输的信息码“1”的个数分别为奇数或偶数，然后在接收端再验证信息码中“1”的个数是否仍为奇数或偶数，以此来判别信息在传输过程中是否出现了差错。具体将在第 6 章介绍。

为便于书写和记忆，有时也将 ASCII 写作十六进制形式（一个字节恰好可以表示两个十六进制），即将某字符的 ASCII 码二进制数形式转换成十六进制数的形式，再标以 H 表示这是一个十六进制的数。例如 A 字母的 ASCII 码为 01000001，写成十六进制即 41H；C 字母的 ASCII 码为 01000011，写成十六进制即 43H。

以上 128 个标准 ASCII 码字符分为两部分：

- 第一部分：由 00H 到 1FH 共 32 个，一般用于通信或作为控制之用，有些可以显示在屏幕上，有些则不能显示，但能看到其效果（如换行、退格）。
- 第二部分：由 20H 到 7FH 共 96 个，这 96 个字符用来表示阿拉伯数字、英文字母大小写和下划线、括号等符号，都可以显示在屏幕上。

这 128 个字符的编码规则如图 1-27 所示，D_n 代表码位。例如大写字母 C 的 ASCII 码，只需在图中对应于字符 C 的位置找出其横坐标 D6D5D4 和纵坐标 D3D2D1D0，依次按 D6D5D4D3D2D1D0 的顺序排列出来，再在最高位补以 0，即得 C 的 ASCII 码为 01000011。

（2）扩展 ASCII 码。

扩展 ASCII 码采用 8 位二进制进行编码，是对标准 ASCII 码的扩展，是由 IBM 制定的，并非标准的 ASCII 码。扩展 ASCII 码由 80H 到 0FFH 共 128 个字符，用来表示框线、音标和其他欧洲非英语系的字母。

$D_3D_2D_1D_0$ \ $D_6D_5D_4$	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	、	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

图 1-27 标准 ASCII 码

2. Unicode 码

Unicode 码最初是 Apple 公司发起制定的通用多文种字符集，后来被 Unicode 协会开发为能表示几乎世界上所有书写语言的字符编码标准。Unicode 码包括 UTF-8、UTF-16 和 UTF-32 这 3 种编码格式，分别指使用 8 位、16 位和 32 位表示字符，在此不作详细介绍。

1.8.2 中文编码

计算机中汉字的表示也是用二进制编码。根据应用目的的不同，汉字编码分为外码、交换码、机内码和字形码。

1. 外码

“外码”也叫输入码，用来解决汉字输入的编码问题，是可用来将汉字输入到计算机中的编码方式。常用的外码（或输入码）有：拼音码、五笔字型码、自然码、表形码、认知码、区位码和电报码等。

2. 交换码

交换码（又称“国标码”）是由中国标准总局于 1981 年制定的中华人民共和国国家标准 GB2312－80《信息交换用汉字编码字符集——基本集》，是在大陆及海外使用简体中文的地区（如新加坡等）强制使用的唯一中文编码。

在 GB2312－80 标准中共收录 6763 个汉字，其中一级汉字 3755 个，二级汉字 3008 个；同时收录了包括拉丁字母、希腊字母、日文平假名及片假名字母、俄语西里尔字母在内的 682 个字符。交换码仅是在计算机内部的一个中文汉字、字符集标准，每个汉字的具体编码实现方式则由下面将要介绍的“区位码”解决。

3. 区位码

“区位码”是国标码的具体编码实现方式。它把国标 GB2312－80 中的汉字、图形符号组成一个 94×94 的方阵，即 94 个区，每个区中又有 94 个位，对应 94 个汉字或字符，总区位数为 8836 个，即总共可编码 8836 个汉字或字符。区位码用来解决计算机识别汉字的能力，也可作为输入码（如区位码输入法），但它并不是计算机内部处理汉字的编码方式，在计算机内部汉字编码是由下面将要介绍的“机内码”进行的。

根据不同区所表示的字符类型可以把整个区位码分成 4 个部分：①01～09 区为特殊符号；②16～55 区为一级汉字，按拼音排序；③56～87 区为二级汉字，按部首/笔画排序；④10～15 区及 88～94 区目前暂未使用，留待以后扩展汉字或字符使用。

4. 机内码

“机内码”是根据“国标码”的规定确定在计算机内部表示一个汉字的二进制代码。在计算机内部和磁盘中汉字代码都用机内码表示。

5. 字形码

字形码是汉字的输出码，是解决汉字输出的编码方式。输出汉字时都采用图形方式，无论汉字的笔画多少，每个汉字都可以写在同样大小的方块中。通常用 16×16 点阵来显示汉字。

1.9 课后自测题

为了巩固本章所学内容，请认真做下面的自测题，参考答案见附录。

1.9.1 填空题

1. 计算机运算中主要有_____、_____、_____和_____4 种数据制式，分别用_____、_____、_____和_____字符标识，它们的基数分别是_____、_____、_____和_____。其中对应 3 位二进制码的制式是_____。

2. 有一个共 12 位的二进制数，如果用 b_n（n 表示位号）表示各个二进制位值的话，整个二进制数可表示为_____，如果把它转换成十进制，则它的计算展开式为_____。如果要把这个 12 位二进制数转换成十六进制，则这个十六进制数共有_____位。假设现有一个从 A4000H 到 CBFFFH 的内存地址段，则这个地址段最多可以存放_____字节数据。

3. 二进制的算术运算中：1+1=_____，0-1=_____，借 1 当_____，进 1 当_____；二进制的逻辑运算中：1∧0=_____，1∨0=_____，1⊕0=_____。

4. 机器数的最高位是_____，其余各位是该数的_____；反码的编码规则是：正数的反码是_____，负数的反码是_____；补码的编码规则是：正数的补码是_____，负数的补码是_____。

5. 机器数 10010110 的真值为_____（用十进制表示），原码为_____，反码为_____，补码为_____（各种编码用二进制数表示）。

6．8 位定点整数 X 的取值范围为_____，8 位定点小数 X 的取值范围为_____。机器数 10110011 的移码是_____。

7．单精度浮点数一共_____位，指数一共_____位，偏移值为_____，双精度浮点数一共_____位，尾数一共_____位，偏移值为_____。

8．中文汉字编码方式有很多种，我们平时使用的 QQ 拼音输入法属于_____，区位码输入法属于_____，在计算机磁盘中表示汉字所采用的是_____，在屏幕上显示汉字所采用的是_____，为汉字输入、输出提供基本汉字、字符集标准的是_____。

1.9.2 选择题（可多选）

1．8 个二进制位至多可表示（　）个数据。

A．8　　B．64　　C．255　　D．256

2．对于 R 进制数，在每一位上的数字可以有（　）种。

A．R/2　　B．R-1　　C．R　　D．R+1

3．假设用 12 个二进制位表示数据，它能表示的最大无符号整数为（　）。

A．2047　　B．2048　　C．4095　　D．4096

4．下列无符号数中最大的是（　）。

A．$(205)_D$　　B．$(001001010010)_B$

C．$(0CD)_H$　　D．$(11100011)_B$

5．与二进制数 100101.001101 等值的十进制数是（　），等值的十六进制数是（　）。

A．25.203125　　B．25.34　　C．37.203125　　D．37.34

6．与十进制数 28.625 等值的十六进制数为（　），与十进制数 254 等值的二进制数是（　）。

A．112.10　　B．1C.A　　C．1C.5　　D．112.5

E．11111110　　F．11101111　　G．11111011　　H．11101110

7．与十六进制数 AC.E 等值的十进制数是（　），等值的八进制数是（　）。

A．112.875　　B．162.875　　C．254.7　　D．172.875

8．与十六进制数 23.4 等值的十进制数为（　），0x1000 转换成十进制是（　）。

A．35.5　　B．23.4　　C．35.75　　D．35.25

E．4096　　F．1024　　G．2048　　H．8192

9．下列关于浮点数说法不正确的是（　）。

A．浮点数中的小数点位置可变

B．浮点数可用一个纯小数和一个指数的乘积表示

C．浮点数的尾数的最高有效位必须为 1

D．浮点数的尾数必须用阶码表示

10．对于两个机器数 55H 和 AAH，运算结果相同的逻辑操作是（　）。

A．逻辑与　　B．逻辑或　　C．逻辑非　　D．逻辑异或

11. 若 X=10111001，Y=11110011，则它们分别进行“逻辑与”和“逻辑异或”运算后的结果分别为（　　）。

A. 110101100，000001101　　B. 001010011，111110010

C. 10110001，01001010　　D. 01001110，11111011

12. 二进制整数采用机器码表示时，（　　）的表示范围最大。

A. 原码　　B. 补码　　C. 反码　　D. 移码

13. 在计算机加减法运算中，最常使用的是（　　）。

A. 原码　　B. 补码　　C. 反码　　D. ASCII 码

14. 反码的作用是（　　）。

A. 作为求补码的中间手段　　B. 作为求原码的中间手段

C. 能将负数转换为正数　　D. 能将减法转化为加法

15. 下面关于补码的作用说法不正确的是（　　）。

A. 使机器数的码制更简单　　B. 使计算机的运算效率更高

C. 能将负数转换为正数进行运算　　D. 能将减法运算转化为加法运算

16. 将-33 以单符号位补码形式存入 8 位寄存器中，寄存器中的内容为（　　）。

A. DFH　　B. A1H　　C. 5FH　　D. DEH

17. 对+0 和-0 表示形式唯一的机器数表示形式是（　　）。

A. 原码　　B. 补码　　C. 反码　　D. 移码

18. 若用 8 位机器码表示十进制数-101，则其原码表示形式为（　　），补码表示形式为（　　）。

A. 11100101　　B. 10011011　　C. 11010101　　D. 11100111

19. 多项式 $2^{14}+2^{11}+2^4+2^1+2^0$ 表示的十六进制数为（　　），表示的十进制数为（　　）。

A. 4813H　　B. 8026H　　C. 2410H　　D. EB410H

E. 18448　　F. 9232　　G. 18451　　H. 36902

20. 设字长 8 位并用定点整数表示，模为 2^8，若$[X]_{补}$=11111010，则 X 的原码及真值分别为（　　）。

A. $[X]_{原}$=00000110，X=+0000110　　B. $[X]_{原}$=10000110，X=-0000110

C. $[X]_{原}$=01111010，X=+1111010　　D. $[X]_{原}$=11111010，X=-0000110

21. 真值 X=-127D，则其 8 位反码及真值分别为（　　）。

A. $[X]_{反}$=11111111，X=-1000000　　B. $[X]_{反}$=10000000，X=-1000000

C. $[X]_{反}$=11111111，X=-1111111　　D. $[X]_{反}$=10000000，X=-1111111

22. 若$[X]_{补}$=CCH，机器字长为 8 位，则$[X/2]_{补}$=（　　）。

A. 34H　　B. 66H　　C. 98H　　D. E6H

1.9.3 计算题

1. 按要求进行下列数制转换（假设全部为无符号数）：

（1）把(01110100)B、(11101001000.10111)B 转换成十进制。

（2）把(1076)O、(6374.65)Q 转换成十进制。

（3）把 0x7A8C、0x259B.25 转换成十进制。

（4）把十进制数 825、10815.6（精确到小数点后 4 位）转换成二进制。

（5）把十进制数 658、9240.65（精确到小数点后 4 位）转换成八进制。

（6）把十进制数 2508、5420.82（精确到小数点后 4 位）转换成十六进制。

（7）把(1011011)B、(10110111.0010)B 转换成八进制。

（8）把(1100111011)B、(11101001.101)B 转换成十六进制。

（9）把(756)O、(6265.42)O 转换成十六进制。

（10）把 0xA58C、0x8152.78 转换成八进制。

2．求下列二进制算术运算结果（假设全部为无符号数）：

（1）求(011101)B +(10010)B、(100111)B +(110110)B 的值。

（2）求(1110101)B -(110010)B、(1101011)B -(10001)B 的值。

（3）求(1110)B ×(1001)B、(1100)B ×(10111)B 的值。

（4）求(110010)B ÷(1010)B、(100110101)B ÷(1011)B 的值。

（5）求(11001)B∧(1011)B、(10011)B∧(10101)B 的结果。

（6）求(11001)B∨(1011)B、(10011)B∨(10101)B 的结果。

（7）求(1100110)B 非和(1000111)B 非的结果。

（8）求(1100110)B⊕(1011)B、(100111)B⊕(10101)B 的结果。

3．求下列补码运算结果（以二进制补码形式表示）：

（1）求$[85+24]_{补}$、$[152+35]_{补}$的值。

（2）求$[185-56]_{补}$、$[52-135]_{补}$的值。

4．求以下浮点数：

（1）求十进制数 20.5 的单精度浮点数。

（2）两浮点数 $X=2^{101}\times0.11011011$，$Y=2^{111}\times(-0.10101100)$，指数和尾数均为二进制原码。现求在存储器中该数的尾数以 9 位补码形式（单符号位），指数以 4 位补码形式（单符号位）的移码的表示形式。

2 计算机网络概述

本章作为本书的开篇（除了作为选学内容的第 1 章），先仅从宏观角度使大家对计算机网络有一个基本的了解。本章内容比较丰富，这对于一个以前对计算机网络一无所知的朋友来说也仍然非常重要。至少你学完本章内容后，在与朋友聊有关计算机网络方面的话题时你也可以对上几句，不会被“冷寞”，不会让你的朋友觉得你完全是一个局外人。

本章主要介绍计算机网络的一些基础知识，包括计算机网络的发展历史、计算机网络的基本组成、计算机网络的作用、计算机网络的分类、计算机网络的拓扑结构等。要着重理解各种拓扑结构的主要特性和优缺点，更加具体、更加复杂的内容将在本书后面各章介绍。

本章主要重点与难点：

- 计算机网络的基本组成
- 计算机网络的分类
- 星型拓扑结构的主要特点
- 环形网络的工作原理
- WLAN 拓扑结构

2.1 计算机网络综述

计算机网络的定义其实并不重要，因为计算机网络自身的内涵也在随着计算机网络新技术、新产品的不断涌现而在不断变化。计算机网络发展至今已有 50 多年的历史了，也没有一个关于计算机网络的定义能完全适合所有不同时期。

目前通常认为“计算机网络”是指将不同地理位置、具有独立功能的多台计算机设备，通过通信线路（包括传输介质和网络设备）连接起来，在网络操作系统、网络管理软件及网络通信协议的管理和协调下实现资源共享、信息传递、网络连接或远程访问等应用目的的计算机系统。至于这些组成部分将在本章后面具体介绍。

2.1.1 计算机网络的发展历史

计算机网络发展至今已经历了几代的变革，不仅计算机网络的内涵发生了巨大的改变，计算机网络技术和网络应用也在发生新的变化。了解计算机网络的整个发展历史，有助于我们对计算机网络技术和应用的发展有一个清晰的认识，也有助于我们分清目前哪些才是主流应用的技术，哪些是已过时的。

计算机网络的主体就是计算机，当然得先有计算机，就像肯定是先有人，然后才有人类社会一样。总体来说，至今为止，可以把整个计算机网络的发展历程分为下面 4 个主要阶段。

1. 第一代计算机网络（1954 年至 1968 年）

1946 年，世界上第一台数字计算机问世，它是电子管计算机。1948 年开发出了晶体管计算机。但当时的计算机数量非常少，而且非常昂贵。由于当时的计算机大都采用批处理方式，所以用户使用计算机，先要将程序和数据抽成纸带或卡片，再送到计算中心去处理。

1954 年，出现了一种称为收发器（Transceiver）的计算机终端，人们使用这种终端首次实现了将穿孔卡片上的数据通过电话线路发送到远地的计算中心计算机。此后，电传打字也可作为远程终端与计算机相连，用户可以在远地的电传打字机上输入自己的程序，而计算中心计算机出来的结果也可以传到远地的电传打字机上并打印出来。这时计算机网络的基本原型就诞生了，是最初的计算机网络，是计算机网络的雏形。

到了 20 世纪 50 年代中后期，通过技术的发展，又可以通过像线路集中器、前端控制器等通信控制设备将地理上分散的多个终端通过公用电话通信网络（PSTN）集中连接到一台计算机主机上，这就是真正意义上的第一代计算机网络，如图 2-1 所示。

以上的“线路集中器”类似于集线器（Hub）和交换机，有多个端口，可以集中连接多台终端设备，而“前端控制器”是为了对计算机主机进行“减负”，因为它可以分担原来由计算机主机完成的数据接收和部分简单的数据处理任务。

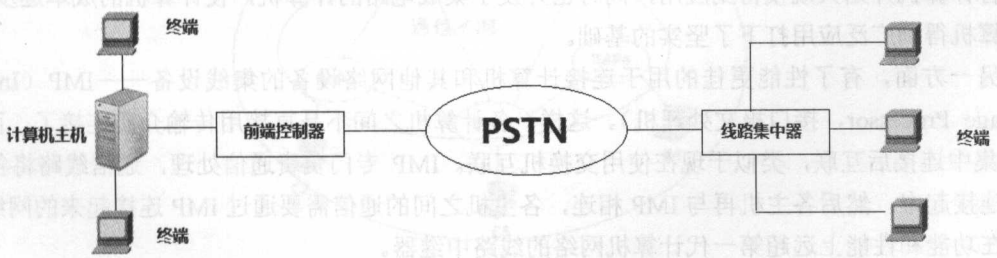


图 2-1 第一代计算机网络基本结构

第一代计算机网络是面向终端的计算机网络，具体特点如下：

- 以计算机主机（相当于“服务器”）为中心，连接着多台终端（不是计算机）。“终端”是仅具有简单数据收发功能的收发器设备，仅包括 CRT 控制器、键盘，没有 CPU 和硬盘，所以没有数据存储和处理能力，类似现在所说的“瘦客户机”。
- 计算机主机承担数据存储和大部分的数据处理任务（前端控制器负责部分简单的数据接收和处理任务），终端不具备数据的存储和处理能力。
- 已可以使用像线路集中器这样的设备集中连接多台终端，简化网络布线，也可以通过 PSTN 进行远程通信。

由于第一代计算机网络是以单个计算机为中心的远程联机系统，所以很显然它具有天生的一些缺点，主要表现在以下几个方面：

- 担当计算中心的计算机主机负荷很重，造成对终端系统的响应比较慢，甚至会出现服务器崩溃现象。
- 单计算机主机系统的可靠性也较低，一旦发生计算机故障，将导致整个计算机网络系统瘫痪。

严格意义上说，第一代计算机网络不能算是真正意义上的计算机网络，因为计算机主机连接的不是计算机，而是不具备独立工作能力的计算机终端。之所以那时网络中更多的是计算机终端，因为那时的计算机非常昂贵。

第一代计算机网络的典型应用是当时的美国航空公司与 IBM 公司在 20 世纪 50 年代初开始联合研究并于 20 世纪 60 年代投入使用的飞机订票系统 SABRE-I。它是由一台计算机主机和全美范围内的 2000 个计算机终端组成。

2. 第二代计算机网络（1969 年至 1983 年）

第二代计算机网络相对于第一代计算机网络来说有着质的飞跃，大大提高了网络的可用性和可靠性，因为它在计算机网络硬件设备、交换技术，甚至计算机网络体系结构上都做了非常多的改进，具体表现如下：

（1）硬件方面的改进。

硬件方面的改进体现在两个方面。一方面是用户使用的是真正意义上的全功能计算机，而不是像在第一代计算机网络中那样，仅具有基本收发功能的计算机终端。因为随着计算机技术的发展，晶体管计算机开始大规模得到应用，同时也开发了集成电路的计算机，使计算机的成本逐步下降，为计算机得到广泛应用打下了坚实的基础。

另一方面，有了性能更佳的用于连接计算机和其他网络设备的集线设备——IMP（Interface Message Processor，接口报文处理机）。这样，各计算机之间不是直接用传输介质连接了，而是由 IMP 集中连接后互联，类似于现在使用交换机互联。IMP 专门负责通信处理，通信线路将各 IMP 相互连接起来，然后各主机再与 IMP 相连，各主机之间的通信需要通过 IMP 连接起来的网络来传送，在功能和性能上远超第一代计算机网络的线路中继器。

（2）交换技术方面的改进。

在第一代计算机网络中，用户计算机终端与计算机主机之间的数据是直接转发的，再加上当时的计算机主机性能很低，数据处理性能差，当连接的终端用户多了以后，就会造成大量用户数据被丢失。为了解决这一问题，各国的科学家们又要寻找新的数据交换技术。

1964 年 8 月巴兰（Baran）在美国兰德（Rand）公司“论分布式通信”的研究报告中提到了“存储－转发”的概念。随后，在 1962 年至 1965 年间，美国的 DARPA（Defense Advanced Research Projects Agency，美国国防部高级研究计划署）和英国的 NPL（National Physics Laboratory，国家物理实验室）都对这一新技术进行了研究。1966 年英国 NPL 的戴维斯（David）首次提出了“分组”（Packet）的概念。1969 年 12 月在美国的 ARPAnet（阿帕网）上应用了这种基于数据分组的交换系统。

【说明】美国的 DARPA 虽然很早就成立了，但直到今天它仍是美国乃至全球最先进、最前沿的国防科技研究机构。世界上许多原来被许多人看作不可实现的技术革命都是由它发明的，如计算机网络中的 Internet、用于全球定位的 GPS、现在全世界各国军队都在大力发展的隐形战机、无人战斗机等。

应用了新的基于分组的“存储－转发”数据交换技术后的 ARPAnet 被认为是第二代计算机网络的正式诞生，也是大家公认的真正意义上的计算机网络，是计算机网络的鼻祖。

（3）体系结构方面的改进。

在第一代计算机网络中，由于网络规模通常非常小，数据都是存储在计算机主机上，所以那时根本没考虑体系结构方面的事。而在第二代计算机网络中，由于各用户使用了真正的计算机主机，可以自己存储和处理数据了，另外引入了性能更强的 IMP 集线设备，所以网络规模就比第一代的大多了。

正因为如此，科学家们开始关注计算机网络体系结构这件事了，以便能更好地发展和维护。在 ARPAnet 中开始把整个计算机网络划分成两部分：各用户计算机主机（包括服务器）划分成“资源子网”，因为网络资源都是存储在这些计算机上的，而用于构建计算机网络通信平台的各 IMP 以及所连接的传输介质共同构成“通信子网”，如图 2-2 所示。用户不仅可以共享“通信子网”中的线路和网络设备资源，还可以共享“资源子网”中其他用户计算机上的丰富硬件/软件资源。

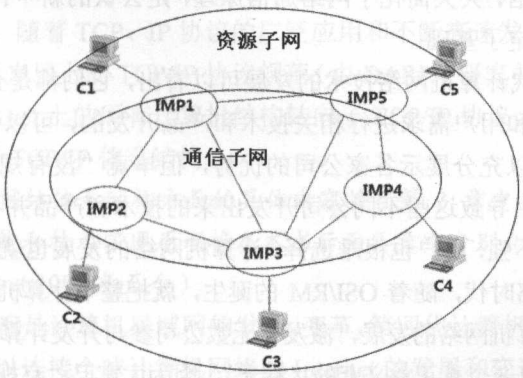


图 2-2　第二代计算机网络

ARPAnet 的成功运行使计算机网络的概念发生了根本性的变化，也标志着计算机网络的发展进入了一个新的纪元。第二代计算机网络的这种既分散（从地理位置上来讲）又统一（从服务功能上来讲）的多主机计算机网络使得整个计算机网络系统性能大大提高，也不会因为单机故障而导致整个网络系统瘫痪。另外，可以使原来第一代计算机网络中的单一计算机主机的负载分散到整个计算机网络的各个计算机主机上，使得计算机网络系统的响应性能也大大提高了。

3. 第三代计算机网络（1984 年至 1991 年）

第三代计算机网络相对第二代计算机网络来说，最主要的改变是体系结构上的重新划分，诞生了国际标准的 OSI/RM（Open Systems Interconnection Reference Model，开放系统互连参考模型）体系结构。OSI/RM 的诞生使得计算机网络体系结构有了一个全球统一的标准，更方便各厂商软/硬件计算机网络产品的兼容，也极大地促进了计算机网络的应用。

尽管第二代计算机网络 ARPAnet 已进行了通信子网和资源子网的两级分层，但这也仅适用于 ARPAnet，其他厂商也在基于自己的产品上设计出不同的计算机网络体系结构，而且这些厂商的体系结构互不兼容，相互不能连接。

如 IBM 于 1974 年推出了系统网络结构（System Network Architecture，SNA），为用户提供能够互联的成套通信产品；1975 年 DEC 公司宣布了自己的数字网络体系结构（DigitaoNetwork Architecture，DNA）；1976 年 UNIVAC 宣布了该公司的分布式通信体系结构（Distributed Communication Architecture，DCA）。

这些计算机网络体系结构只适用于由对应公司产品组成的计算机网络，采用不同公司网络设备组成的计算机网络间仍不能相互连通。这种各自为政的状况使得用户在投资方向上无所适从，也不利于多厂商之间的公平竞争，更不利于整个计算机网络的发展。

针对上述情况，1977 年 ISO（国际标准化组织）的 TC97 信息处理系统技术委员会 SC16 分技术委员会开始着手制定开放系统互连参考模型（OSI/RM），并于 1984 年发布。OSI/RM 模型是一个开放体系结构，定义了网络互连的七层结构，并详细规定了每一层的功能（如图 2-3 所示），以实现开放系统环境中的互连性、互操作性和应用的可移植性。OSI/RM 模型同时规定了计算机之间只能在对应层之间进行通信，大大简化了网络通信原理，是公认的新一代计算机网络体系结构的基础，为普及计算机网络奠定了基础。

从以上第一代、第二代计算机网络技术的发展可以看出，它们都是企业驱动的，也就是由各个公司自己根据自己的市场和用户需求进行相关技术和产品开发的，可以说是处于“百家争鸣”的时代。虽说“百家争鸣”可以充分展示各家公司的优势，但毕竟“没有规矩难成方圆”。由于这些公司间彼此的市场竞争关系，导致这些不同公司开发出来的技术和产品并不具有通用性，这样一来的结果就是谁都做不大、做不强，用户也很难选择，计算机网络的发展也就很难真正有实质性的进展。

到了第三代计算机网络时代，随着 OSI/RM 的诞生，就把整个计算机网络体系结构以标准形式确定下来了，大大推动了计算机网络的发展，激发了无数公司参与开发计算机网络相关硬件和软件，迎来计算机网络发展历史上第一个真正意义上的“春天”，因为大家已“有规可循”了，不再担心兼容问题了。当然 OSI/RM 标准的诞生也是在汇总了前面不同公司所开发的体系结构优点的基础上开发的。

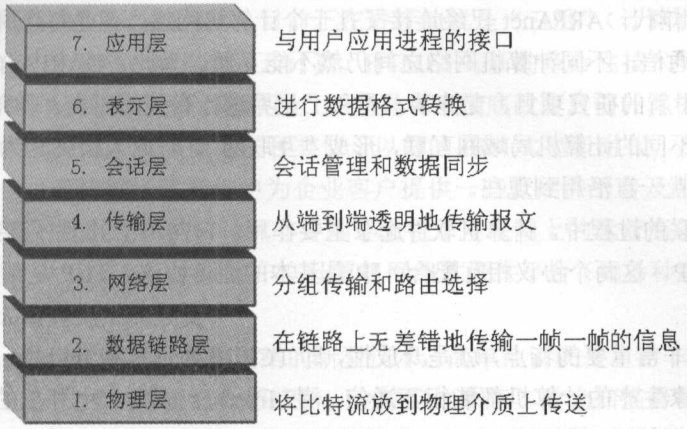


图 2-3 OSI/RM 的七层结构及对应的基本功能

OSI/RM 的诞生也标志着第三代计算机网络的诞生。此时的计算机网络在共同遵循 OSI 标准的基础上形成了一个具有统一计算机网络体系结构并遵循国际标准的开放式和标准化的网络。1980 年 2 月，IEEE 学会下属的 802 局域网标准委员会宣告成立，并相继推出了若干个 802 局域网协议标准，其中绝大部分后来被 OSI 正式认可并成为局域网的国际标准。它标志着局域网协议及标准化工作向前迈出了一大步。

从 1980 年至今，802 委员会已陆续发布了环形网、总线网、令牌总线网、光纤网、宽带网、城域网和无线局域网等许多局域网标准。这些 IEEE 802 局域网标准的制定极大地推进了计算机局域网的发展。近几年，计算机局域网的发展速度更是惊人，千兆、万兆、4 万兆、10 万兆等局域网技术也已进入应用阶段。不仅如此，以太网的传输距离已从原来局域网的范围延伸到了城域网、广域网的范围。正因为如此，现在的局域网和广域网之间的界限变得越来越模糊了。事实上，现在不仅我们几乎每个单位的计算机网络都连接上了 Internet，而且我们每个人的计算机都已接入了 Internet，成为了 Internet 的一分子。

最后不得不说的是，虽然 OSI/RM 的诞生大大促进了计算机网络的发展，但主要还是在局域网范围中，在后来的广域网，随着 TCP、IP 协议的广泛应用和不断高速发展，包括 Internet（互联网）的应用中，OSI/RM 却被后来居上的 TCP/IP 协议规范（由 DARPA 研究并发布）远远地抛在了后面。1983 年，DARPA 将 ARPAnet 上的所有计算机结构转向了 TCP/IP 协议，并以 ARPAnet 为主干建立和发展了 Internet，形成了 TCP/IP 体系结构。

【说明】有关计算机网络体系结构方面的具体内容将在第 3 章中具体介绍，有关计算机网络体系结构中各层的主要功能和技术实现原理将在本书后面各章中分别介绍。

4. 第四代计算机网络（1992 年至今）

前面三代可以说主要都是计算机局域网的发展、变革，第四代计算机网络则主要聚焦于广域网，特别是最大的广域网、可以连接全球计算机网络的 Internet 的发展和变革。当然这并不是说前面三代只针对局域网，因为事实上后面发展的 Internet 也是基于第二代计算机网络 ARPAnet。

在 20 世纪 70 年代，ARPAnet 已经连接了几十个计算机网络，但是每个网络只能在网络内部的计算机之间互联通信，不同计算机网络之间仍然不能互通。为此，DARPA（美国国防部高级研究计划局）又设立了新的研究项目，支持学术界和工业界进行有关的研究。研究的主要内容就是想用一种新的方法将不同的计算机局域网互联，形成“互联网”。研究人员称之为 Internetwork，简称 Internet。这个名词就一直沿用到现在。

在研究实现互联的过程中，计算机软件起了主要作用。1974 年，出现了连接分组网络的协议，其中就包括 TCP/IPP。这两个协议相互配合，IP 是基本的通信协议，TCP 是帮助 IP 实现可靠传输的协议。

TCP/IP 有一个非常重要的特点，就是开放性，即 TCP/IP 的规范和 Internet 的技术都是公开的。目的就是使任何厂家生产的计算机都能相互通信，使 Internet 成为一个开放的系统。这正是后来 Internet 得到飞速发展的重要原因。

DARPA 在 1982 年接受了 TCP/IP，选定 Internet 为主要的计算机通信系统，并把其他的军用计算机网络都转换到 TCP/IP。1983 年，ARPAnet 分成两部分：一部分军用，称为 MILnet；另一部分仍称 ARPAnet，供民用。

1986 年，美国国家科学基金组织（NSF）将分布在美国各地的 5 个为科研教育服务的超级计算机中心互联，并支持地区网络，形成 NSFnet。1988 年，NSFnet 替代 ARPAnet 成为 Internet 的主干网。NSFnet 主干网利用了在 ARPAnet 中已证明是非常成功的 TCP/IP 技术，准许各大学、政府或私人科研机构的网络加入。1989 年，ARPAnet 解散，Internet 正式从军用转向民用。

Internet 的发展引起了商家的极大兴趣。1992 年，美国的 IBM、MCI、MERIT 三家公司联合组建了一个高级网络服务公司（ANS），建立了一个新的网络，叫做 ANSnet，成为 Internet 的另一个主干网。它与 NSFnet 不同，NSFnet 是由国家出资建立的，而 ANSnet 则是 ANS 公司所有，从而使 Internet 开始走向商业化。

随着 Internet 的发展，经过 ANSnet 商业化动作后很快就覆盖了全球 91 个国家，主机已超过 400 万台。于是，1995 年 4 月 30 日，NSFnet 正式宣布停止运作。再经过不断发展，就成了现在几乎无所不连，无所不包的 Internet 了。

5. 下一代计算机网络

有人问，我们现在的计算机网络处于一个什么时代，可以这么说，我们目前正处于第三代、第四代，以及下一代之间的过渡时期。因为第三代的计算机网络体系结构、第四代的 Internet 仍在不断完善中，所包括的各种局域网、广域网技术和标准也在不断发展中，新的计算机网络技术、产品和应用也是层出不穷，所以不能把现在简单地看成是处于某一代。

总体而言，下一代计算机网络（NGN，也就是通常所说的第五代计算机网络）普遍认为是因特网、移动通信网络、固定电话通信网络的融合，IP 网络和光网络的融合；是可以提供包括语音、数据和多媒体等各种业务的综合开放的网络架构；是业务驱动、业务与呼叫控制分离、呼叫与承载分离的网络；是基于统一协议的、基于分组的网络。

在功能上，NGN 分为四层，即接入和传输层、媒体层、控制层、网络服务层。我们看得见的

一些下一代计算机网络的主要特征包括：目前正在进行的就是“三网”（计算机网络、电信网络、广播电视网络）融合，物联网、虚拟化、云计算、HTML5 等新的革命性技术等。

在这些新技术中，“云计算”和“物联网”可能是将来彻底改变目前计算机网络格局和应用的两大最主要的技术。“云计算”其实就是一种类似于以前的 IBM 大型机时代，是一种集中服务、集中管理的平台。就是由云计算运营商集中为企业客户提供一些软硬件平台及各种所需的服务和管理，企业客户只需要通过比较简单的云计算客户端连上位于互联网上的运营商云计算平台即可享受所购买的服务、平台，大大节省了企业客户在计算机网络软硬件平台（如各种服务器系统、企业交换机、路由器、防火墙等）上的投资。

“物联网”是一种继续扩展计算机网络的新型技术，简单地讲就是“物－物相联的网”。它是通过射频识别（RFID）技术，以及红外感应器、全球定位系统、激光扫描器等信息传感设备，按约定的协议，把一些目前不能与计算机网络连接的物品（如电灯、电器、监控设施等）与互联网连接，以便进行物品之间的信息交换和通信，实现对物品的智能化识别、定位、跟踪、监控和管理。通过“物联网”我们将来在上班时就可以控制家里的电灯、电器设施的开关，监控家里的防盗监控设施，真正实现无处不在的互联网应用。

2.1.2 计算机网络的基本组成

无论是从以上哪个定义都可以看出计算机网络是一个由一些硬件设备和相应的软件系统组成的。其基本组成包括：计算机设备、网络设备、传输介质、通信软件。总体来说就包括：计算机网络硬件系统和计算机网络软件系统两大部分。

1. 计算机网络硬件系统

计算机网络硬件系统是指计算机网络中所有可以看得见、用于计算机网络连接和通信的物理设施，如各种计算机设备、传输介质、网络设备这三大部分。

（1）计算机设备。

组建计算机网络的目的就是为各种计算机设备用户之间的网络通信（可以是用户访问、数据传输、文件共享、远程控制等各种应用）提供平台。计算机设备就是由网络用户所控制和使用的各种计算机（如 PC、计算机服务器、计算机终端、笔记本电脑、iPAD 等之类的便携式设备）。

【说明】在传统的计算机网络定义中，计算机网络至少要求有一台完整功能的物理计算机（其他的可以是终端），随着网络虚拟化技术的兴起，目前的计算机网络可以通过虚拟机软件（如 VPC、VMware 等）在一台物理计算机中模拟多台独立计算机系统，组成一个虚拟的计算机网络，同样可以实现许多物理计算机网络中所能实现的功能。

（2）网络设备。

网络设备是用来构建“通信子网”中网络拓扑结构的，与所用的通信线路（即“传输介质”）一起共同构成整个计算机网络的骨架。当然最简单的网络其实是可以不需要任何网络设备的，那就是两台终端计算机用串/并口电缆直接连接起来的对等网络。但这种网络并不能算是真正意义上的计算机网络，现在来说，这样的计算机网络也没有多大实际意义。

在计算机网络系统中，网络设备通常是指除计算机设备以外的设备，更为复杂，因为为了实现网络的连接，需要能实现不同功能的设备。

最基本的网络设备就是安装在每台 PC、服务器上的网卡（包括有线网卡和无线网卡），还有用于连接这些计算机的交换机（早期的中继器、网桥、集线器现在基本不用了）、无线 AP（接入点），用于连接不同网络的设备有网关、路由器，用于提供安全防护的设备有防火墙、IDS（入侵检测系统）、IPS（入侵防御系统），用于远程广域网连接的各种 Modem、宽带接入服务器（BRAS）、光端机（也称光纤收发器）等。

（3）传输介质。

“传输介质”简单地说就是网线，用于构建计算机网络设备间通信的“路”。如果没有这些传输介质，网络信号将不知道往哪里传，也不可能传，就像前面没有路，我们无法向前行进一样。

传输介质可以是物理有形的，如同轴电缆（有线电视所用的电缆也是同轴电缆）、双绞线、光缆（也叫光纤）等（如图 2-4 所示）；还可以是无形的，如各种无线网络中使用的传输介质其实就是各种电磁波，无线计算机网络就是通过电磁波实现无线计算机网络中各节点的连接的。当然，在同轴电缆、双绞线、光缆这些传输介质中又有许多分类，具体将在第 5 章中专门介绍。

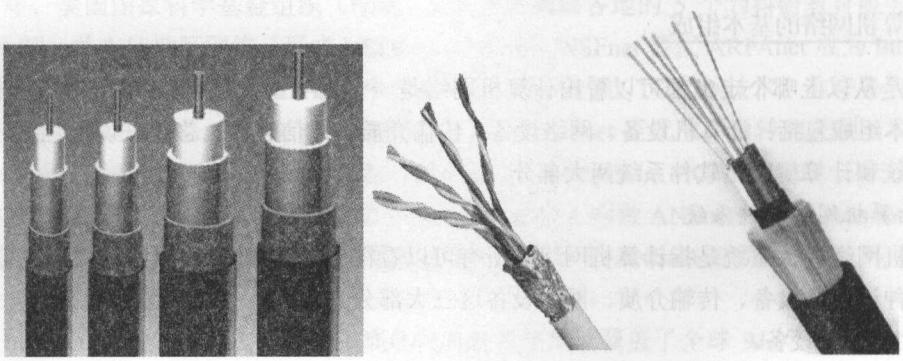


图 2-4　同轴电缆、双绞线、光缆

2. 计算机网络软件系统

仅有以上这些硬件设备是不可能组成计算机网络的，还必须有相应的软件系统支持。包括安装在 PC、服务器、交换机、路由器、防火墙等所有设备上的网络操作系统，如安装在 PC 和服务器上的 Windows、Linux、UNIX 等网络操作系统，安装在 Cisco 交换机、路由器上的 IOS 系统，安装在华为设备上的 VRP 系统等；还包括要实现某项网络应用目的的网络应用软件，如即时通信类软件（如 QQ、MSN 等）、文件传输 FTP 协议类软件（如 Serv-U、Easy FTP、TFTP 服务器软件等）、电子邮件通信类软件（如 Outlook、Foxmail、Gmail 等）、浏览器访问软件（如 IE、Chrome、Firefox 等浏览器）、各种拨号软件等。

除此之外，计算机网络软件还包括独立或者内植于操作系统中的网络通信协议，如 TCP/IP 协议簇、IEEE 802 协议簇等，网络设备中的 VLAN、STP、RIP、OSPF、BGP，拨号通信的 PPP、PPPoE

协议，VPN 通信的 IPSec、PPTP、L2TP 协议等。

2.1.3 计算机网络的主要应用

时至今日，如果要谈计算机网络的应用那可以说随处可见了，最常见的就是我们每个人每天都在进行的 Internet 访问，如网站的访问、网盘的使用、文件传输、电子邮件发送、QQ 聊天等，已渗透到了普通百姓的日常工作、生活、娱乐和休闲等各个方面。如果从大的分类来看，计算机网络应用主要包括：资源共享、文件传输、用户访问、网络互联、协同工作等几个方面。

1. 资源共享

“资源共享”是构建计算机网络最基本也是最重要的一个目的。这里所说的“资源共享”包括：硬件资源共享、软件资源共享、文件资源共享和数据资源共享这 4 个方面。

硬件资源共享最典型的例子就是我们最常用的打印机共享（如图 2-5 所示）、扫描仪和磁盘共享（如网吧中的无盘网络中）。在网络设备中，各业务网板、模块共享同一块主控板也是一种硬件资源共享。

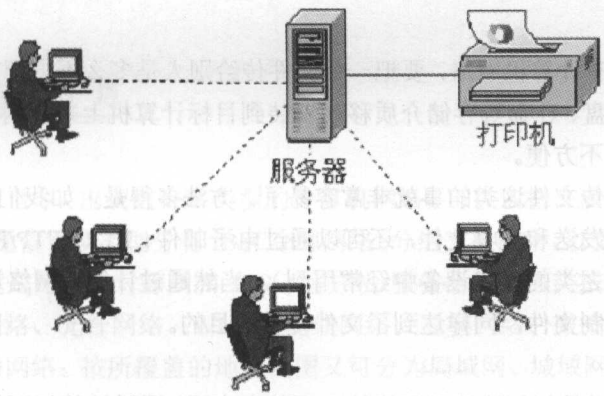


图 2-5 硬件资源共享示例

软件资源共享最常见的就是我们许多用户可以共享同一个服务器系统，如我们经常玩的网络游戏，各用户就是共享了安装在服务器中的游戏软件。在各种操作系统虚拟化应用中（如 VMware、VPC 系统），多个虚拟主机也可以实现与物理主机共享软硬件资源的共享。在当今云计算时代，基于软件的云就可以在云中心统一为许多用户提供统一的各种软件应用平台共享，如各种 Office 办公云系统。

文件资源共享就更常见了，如通过 Internet 中的网盘为全球用户提供各种各样的资源，各个下载类的网站为全球用户提供一些软件的下载。笔者也是通过网盘为所有会员提供视频课程及工具软件下载的，比起一个个用户单独传输，那是方便多了。当然，在企业中也可以通过构建私有或者开放的各种文件服务器来实现文件资源共享。我们访问各大网站也是一种文件资源共享，可以在网站上打开一篇篇文章，获取各方面所需的信息。

数据资源共享最典型的例子就是数据库资源共享，公司网络用户甚至公司客户、合作伙伴均可以根据权限查看公司数据库服务器中的相关数据信息，如图 2-6 所示。当前各种云计算中心也可以实现所有用户根据需要查阅所需的数据，实现数据资源共享。

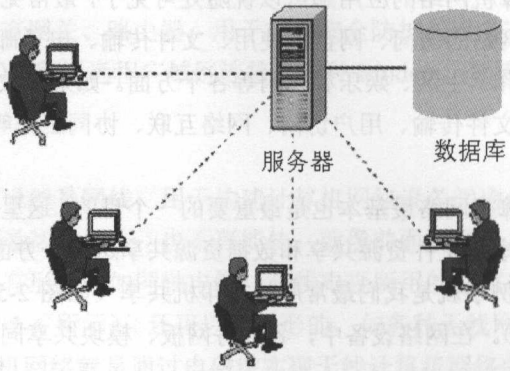


图 2-6　数据库资源共享示例

2. 文件传输

试想一下，如果没有计算机网络，要把一个文件传给别人是多么不方便。只能通过把安装在一台计算机上的磁盘、光盘、U 盘等存储介质移动安装到目标计算机上来复制了。这样，即使在同一个家庭中都会感到十分不方便。

有了计算机网络，传文件这类的事就非常容易了。方法多得是，如我们天天通过 QQ、微信之类的即时通信工具软件发送和接收文件，还可以通过电子邮件、FTP/TFTP/RCP 协议进行文件传输（这在交换机、路由器之类的网络设备中经常用到）。当然通过计算机网络还可以在各网络用户计算机之间直接移动、复制文件，同样达到了文件传输的目的。

3. 用户访问

“用户访问”可以说是任何计算机网络的一项基本应用，通过计算机网络可以访问网络中的其他用户，然后再进行其他所需的网络应用，如资源共享、文件传输、文字/视频聊天、电话通信等。

为了方便网络中用户间的相互访问，通常采用的是 C/S（客户端/服务器）工作模式，用户账户统一建立在服务器上，由服务器进行统一管理和授权，这样同一网络中的计算机用户之间就可以根据所赋予的权限进行相应的访问和网络应用了。

4. 网络互联

“网络互联”就是把不同网络通过一些解决方案连接起来，主要是进行远程的网络互联。如位于不同地点的集团分机构网络与集团总部网络，或者各分支机构之间的网络进行互联。在解决方案上也可以根据不同需要采用不同的方式，如可以通过专线连接，也可以通过拨号远程连接，目前主要是通过各种 VPN（虚拟专用网）通过公用的 Internet 实现跨地区或者跨国家地进行网络互联。

其实“网络互联”的最终目的还是要在不同的网络中实现资源共享和用户访问，当然也可以是其他各方面的网络应用，因为只要网络通了，其他网络应用就可以自由根据需求选择了。远程视频

会议、远程培训、远程会诊等的应用目前在一些大的集团公司或者医院中是最常见的。一方面可以节省会议、培训成本（因为它克服了物理距离的时空限制），另一方面也可以充分利用各方面的专家资源及时地解决一些疑难杂症。

5. 协同工作

“协同工作”是目前计算机网络的一种新型应用，也称“分布式应用”。它是指通过计算机网络，使位于相同或者不同地点，甚至不同国家的人员共同参与完成某个项目、工程任务。

如某跨国公司的设计部门就可能需要位于不同国家的分设计部门的人员来参与集团公司某项重要的科研、开发和设计任务，所有工作任务的分派、技术讨论、进程协商等主要通过网络进行。如果没有计算机网络，这样复杂的工程项目是很难让分布于全球的员工来很好地参与进来。

以上列举的只是一些典型的计算机网络应用领域，随着计算机网络技术、产品的发展，新的计算机网络应用也会不断涌现。总之，凡是可以利用计算机网络做的事都是计算机网络的应用范围。如未来物联网时代，我们通过一个智能终端就可以管理一切了（如各种家电开关、防盗、防火等），这时的计算机网络应用范畴就更广了。

2.2 计算机网络的分类

自计算机网络诞生至今，出现过许多类型的计算机网络，可以依据许多不同的分类标准来划分这些计算机网络。如按通信协议类型分有 IBM 令牌网络、分组交换网络、以太网协议局域网、TCP/IP 协议网络等；按管理模式可分为对等网和 C/S（客户机/服务器）网络；按不同传输介质可分为同轴电缆网络、双绞线网络、光纤网络、WLAN 无线网络、卫星网络、微波网络等；按传输方式又分为点对点网络和广播网络；按所覆盖的地理范围又可分为局域网、城域网和广域网等。下面介绍几种目前最常见的计算机网络分类方式。

2.2.1 按网络所覆盖的地理范围分

按照计算机网络分布的地理范围（也是一种按网络规模进行的计算机网络分类）可分为局域网、城域网和广域网 3 种。这种分类方式可以很好地反映不同类型网络的技术特征，因为不同类型网络覆盖的地理范围不同，所采用的传输技术也就不同，从而形成了不同的网络技术特点与网络服务功能。

1. 局域网 LAN（Local Area Network）

局域网（LAN）是我们最常见到的一种网络类型，也是最多的一种计算机网络，大到各行各业的企业内部网络，小到千家万户的家庭网络都属于局域网（仅指内部网络部分）。我们常说的校园网通常也是一种局域网。局域网是将一个比较小区域内的各种通信设备互连在一起的计算机网络。

局域网具有如下几个方面的主要特点：

（1）私有服务。

局域网属于个人或单位自建，所以其用途也是完全出于私用，不会为网外人员提供服务，如企业局域网通常只对本单位员工提供服务。在局域网中也是采用专门为局域网分配的私有 IP 地址（这种地址每个公司和个人都可直接拿来使用，无需购买，也无需注册）。

（2）分布范围较小。

局域网中各计算机网络设备分布的地理范围较小，小的可以只是在自己家里那几平方米范围内，当然也可以是相距较远范围内的设备相连，如分布在公司中的不同建筑物中。局域网的地理分布范围通常最大只是在 10 公里范围内，但可以分布在不同地理位置的建筑物内。

（3）结构简单，布线容易。

因为局域网都为个人或单位私用，所以网络结构相对较为简单，没有也无需太多、太复杂的网络设备和应用，只需满足自身的网络应用需求即可。同时大多数局域网都采用比较廉价的双绞线布线（当然，根据需要也可以采用光纤作为传输介质），而且因为分布范围比较小，所以布线方式较为简单，容易实现。

（4）网络速度较快。

目前以太网局域网技术的发展非常迅速，最快的以太网速率已达到 100Gb/s，相对广域网和互联网来说具有非常大的优势。这也为企业局域网的集中应用提供了保证。

（5）误码率低。

因为局域网的结构相对比较简单，连接带宽高，传输距离短，所以通信的误码率都比较低，通常在 10^{-8}～10^{-11} 之间。

局域网中的类型和规范非常多，如目前最主流的以太网（EtherNet）、WLAN（无线局域网），以前还有 IBM 令牌网（传输速率最高仅为 16Mb/s）、FDDI（光纤分布式数据接口）网（最高传输速率可达 100Mb/s）、ATM（异步传输模式）网（最高传输速率可达 Gb/s 以上）。而在最常见的以太网中又有许多规范，如 10Mb/s 的标准以太网、100Mb/s 的快速以太网、1Gb/s 的千兆以太网、10Gb/s 的万兆以太网和 100Gb/s 的十万兆以太网。有关以太局域网规范将在第 7 章介绍，有关 WLAN 接入规范将在第 5 章介绍。

2. 城域网 MAN（Metropolitan Area Network）

城域网（MAN）中各计算机网络设备的地理分布范围介于局域网和下面将要介绍的广域网（WAN）之间，主要是分布在一个城市中，所以称之为“城域网”。城域网主要是用来在一个较大的地理区域（通常是 10～100 公里之间）内提供数据、声音和图像的传输，一般是用于提供公共服务的。

MAN 的标准为 IEEE 802.6，通常采用 ATM 技术作为骨干网传输技术，不过目前光纤技术也在城域网中得到了广泛应用。ATM 是可同时应用于数据、语音、视频和其他多媒体应用的一项网络传输技术。城域网通常是为一个或几个组织所有，更多的是为公众提供公共服务，如城市银行系统、城市消防系统、城市邮政系统、城市有线电视/广播网络等。

3. 广域网 WAN（Wide Area Network）

广域网（WAN）是规模最大的一种计算机网络，分布的地理范围可以非常广，如一个或多个城市，或者多个国家，甚至可以遍布全球，如 Internet。

一般的广域网是一些比较大的集团公司、政府组织或者运营商（ISP）构建的，它们连接了分布于全国，甚至全球的多个局域网，如一个跨国公司就需要连接分布于多个国家的分支机构的局域网。政府组织中的广域网就更多了，如我国分布在各省、市、县甚至乡镇的政务网、教育网、医疗卫生网、公安网等。各 ISP 构建的广域网主要是为公众提供公共服务，如现在的电话通信网、移动通信网、专线接入网等。

广域网具有以下基本特点：

（1）覆盖范围广。

广域网所覆盖的地理范围可以非常广，一般从几百公里到几千公里，也可覆盖多个城市、整个国家，乃至全球。Internet 是最大的广域网，其他的广域网又是 Internet 的核心。在广域网中通常使用的是公网 IP 地址（这种地址是需要购买并注册的），当然广域网互联的各局域网内部仍可使用供局域网私用的 IP 地址。

（2）构建成本高。

由于广域网地理位置分布广，网络线路很长，介质类型多种多样，而且铺设非常困难，所以单独组建一个广域网的成本非常昂贵，所以通常是借用传统的公共传输网这个平台来实现。当然也可租用 ISP 提供的各种专线。对于 ISP 来说，广域网的构建一般都是通过光纤专线连接的，以确保网络连接性能。

（3）网络结构和类型复杂。

由于广域网连接了多个远程网络，所以网络结构非常复杂，网络类型也可能不一样，需要解决不同结构和不同类型网络之间的互联。

（4）传输速率低，误码率高。

由于广域网的传输距离远，又依靠传统的公共传输网，带宽有限，所以误码率较高（一般在 10^{-7}～10^{-8} 之间），传输速率较低（通常是 100Mb/s 以内）。

图 2-7 显示了局域网、城域网和广域网三者之间的典型关系。其中的“接入网”是位于局域网与城域网或广域网之间的中间小型网络，专门为远程网络间互联提供网络用户接入技术，如各种 ISP（Intenert 服务商）为我们提供的 ADSL 拨号、光纤接入、分组接入、卫星接入，以及各种专线接入等。

2.2.2 按网络管理模式分

按计算机网络的管理模式可以把目前的计算机网络划分为对等网（Peer-to-Peer，PTP）和 C/S（Clent/Server，客户机/服务器）网。

1. 对等网

所谓“对等网”（Peer to Peer）就是网络中各成员计算机的地位都是平等的，没有管理与被管

理之分，各自为政，谁也不管谁，采用的是分散管理模式。

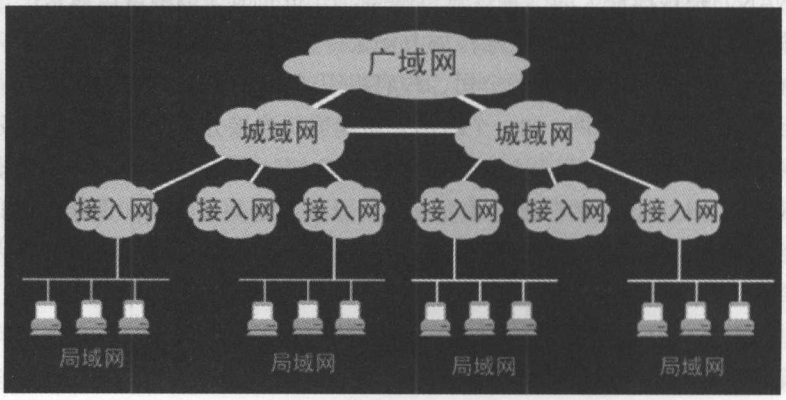


图 2-7　局域网、城域网和广域网之间的关系

就像日常的一些小沙龙一样，沙龙中各成员是平等的，大家组织在一起仅是为了相互交流。对等网中的每台计算机都既可以作为其他计算机资源访问的服务器，同时每台计算机又可作为工作站来访问其他计算机，整个网络中没有专门的资源服务器，如图 2-8 所示。最简单的对等网可以仅通过串行电缆（称为“零调制解调器”）来连接两台计算机。

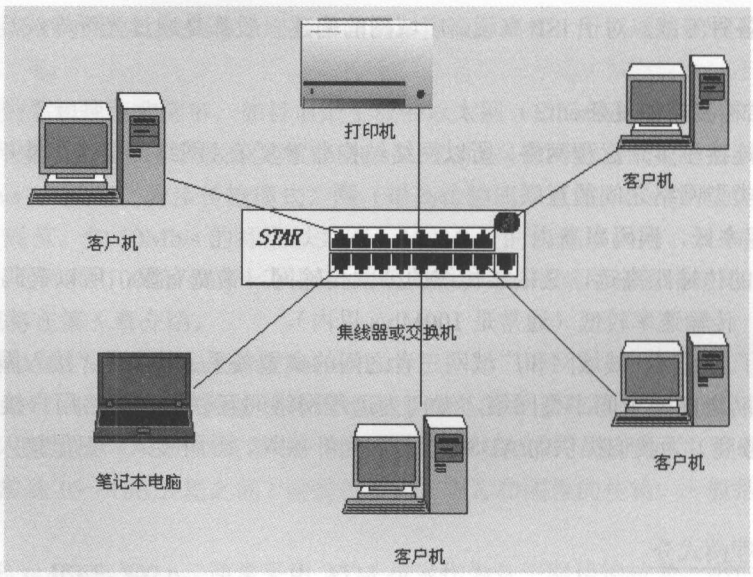


图 2-8　对等网示例

对等网可以说是当今最简单的网络，远没有像 Windows 域网络那样的 C/S 网络配置复杂，非常适合家庭、校园和小型办公室用户。从用户和计算机管理角度来看，Linux 和 UNIX 操作系统组成的计算机网络中都是采用这种网络管理模式，Windows 操作系统中的“工作组”网络也是对等

网管理模式。但需要注意的是，即使在对等网中，也可能有部分服务是采用 C/S 管理模式的，如在工作组网络中部署了文件服务器、数据库存服务器、邮件服务器等。

对等网在具有配置简单的优点之外，更多的是它的不足，主要体现在以下几个方面：

（1）配置与管理困难。

因为采用的是分散管理模式，所以对等网络管理员无法做到通过一台计算机集中管理网络中的用户、计算机和其他资源，更难以对网络中的用户计算机进行统一配置。

（2）安全性差。

同样是因为对等网采用分散管理模式，给整个网络（特别是企业网络）的安全性带来了巨大挑战。因为在这种计算机网络中，企业数据是散在各用户计算机上保存的，很难十分有效地为每一用户计算机采取高级的安全保护措施。

（3）成本高。

表面上看对等网不需要服务器，成本会更低，事实上不完全是这样的，特别是对于有一定规模的企业网络。因为企业数据是分散在各用户计算机上存储和管理的，随着时间的推移，可能需要为每个用户计算机配备较大容量的磁盘，浪费了磁盘空间。再有如果要为每个用户配备相应的安全保护措施，这成本就更高了。

（4）性能差。

在对等网中，各用户计算机的网络连接性能、数据处理性能和磁盘读取性能基本上一样，而且通常只是普通的性能，再加上各用户计算机之间可能因一些数据共享而需要频繁访问，所以对等网的网络通信和数据读取性能就只能一般了。而下面将要介绍的 C/S 模式网络中，服务器的网络连接性能、数据处理性能和磁盘读取性能（通常是采用服务器专用磁盘）都比较高，而且用户间的访问通常比较少，所以 C/S 模式的整体网络性能往往会更高。

2. C/S 网络

C/S 模式其实是针对具体服务器功能来说的，这些服务器可以是用以管理整个计算机网络中计算机和用户账户的服务器（如 Windows 域网络中的域控制器），也可以是其他网络或者应用服务器（如邮件服务器、数据库服务器、Web 服务器、FTP 服务器等）。这些服务器有一个共同的特点就是一般只作为服务器角色而存在，专门为网络中的其他用户计算机提供对应的服务。这相当于我们现实生活中每个组织、每个部门都有一个或多个管理者一样，他们负责的就是对对应的组织或部门进行管理。如图 2-9 所示是一个单台服务器的 C/S 网络，而图 2-10 所示是一个具有多台不同类型服务器的 C/S 网络。

C/S 网络的杰出代表就是 Windows 服务器系统（如 Windows Serever 2008、Windows Serever 2012 等）的域网络。这种域网络可以对网络中的所有用户、计算机账户进行统一管理。除此之外，各种其他应用层网络服务器（如 DNS 服务器、DHCP 服务器、NFS 服务器等）和应用服务器（如 Web 服务器、FTP 服务器、E-mail 服务器等）也可以组成 C/S 模式网络。

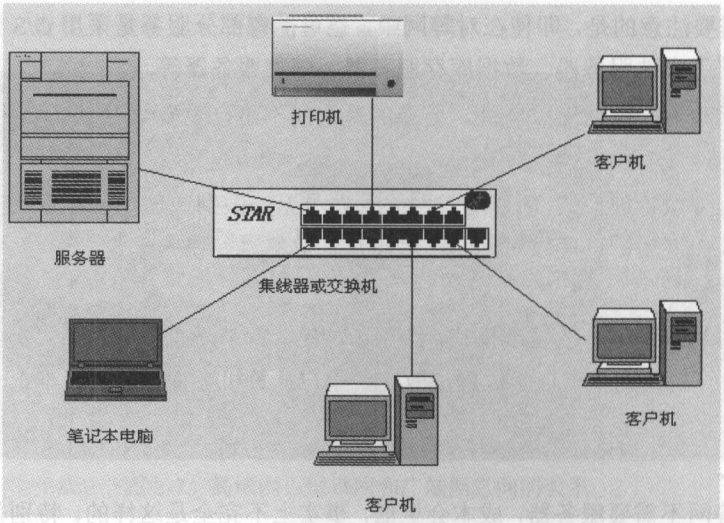


图 2-9　单服务器的 C/S 网络

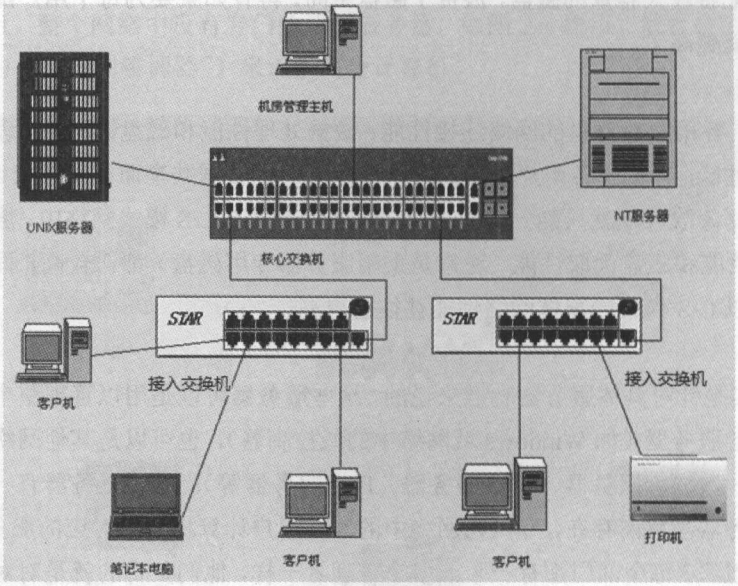


图 2-10　多服务器的 C/S 网络

综合起来，C/S 模式网络的主要优点如下：

（1）管理和配置容易。

在 C/S 模式网络中，往往用户账户可以集中管理，只需在服务器上进行一些稍微复杂的配置，在客户的配置比较简单。如对于 Windows 域网络，可以集中管理整个网络中的用户/计算机账户和安全策略。

（2）安全性高。

同样是因为在服务器上集中管理了用户账户、计算机账户、企业数据（重要的数据一般都要求直接保存在服务器上，不允许保存在用户计算机中），以及整个网络的安全策略，所以整个网络的安全性较高，也更方便部署。

（3）性能好。

因为网络中有专门的高性能服务器，同时分配有高性能的交换处理设备、高带宽端口和高性能磁盘（通常还使用性能和安全性更高的磁盘阵列），所以整个网络中尽管用户集中访问服务器会比较频繁，但就整体网络性能来讲，仍比对等网模式要高。

至于C/S模式网络的不足主要可能还是对于小型企业网络来说成本有些高，因为要专门配备高性能的服务器。而且还存在单点故障风险，因为整个网络中由一台或少数几台服务器进行统一的用户/计算机账户、数据资源和安全策略的管理，一旦这些服务器出现故障，整个网络也就瘫痪了。这时通常采用服务器集群来实现热备份。

2.2.3 按传输方式分

按计算机网络的传输方式划分为：广播（Broadcast）网络、点对点（P2P）网络、点对多点（P2MP）网络和非广播多路访问（NBMA）网络这 4 种。其实在企业网络中只有前面两种，后面的 P2MP 和 NBMA 两种网络类型仅在广域网中存在。

1. 广播网络

广播（Broadcast）网络是可以在公共信道以广播方式传输数据的一种计算机网络，拓扑结构带有总线型特性（各用户主机间通信存在公共信道）。以太网络就是一种典型的广播网络。如图 2-11 所示就是一个广播式传输网络示例，图中 PC1 发送一个广播包（如 ARP 请求包），可以到达网络中其他任何 PC 上。

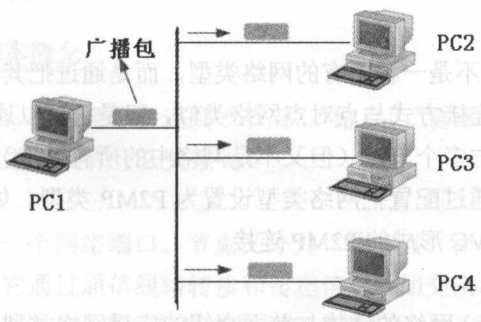


图 2-11 广播式传输网络示例

在广播网络中传输信息时，任何一个节点都可以通过公共信道（如交换机的背板矩阵以及设备间的连接通道）或总线以广播方式传送数据到网络中的其他每台计算机上。然后，这些计算机根据数据包中的目的 MAC 地址进行判断，如果自己的 MAC 地址与目的 MAC 地址匹配则接收（与此

同时，发送节点就可以知道对应目的 IP 地址所对应的 MAC 地址，下次接收到包括同样目的 IP 地址的包时就不用再广播了），否则便丢弃它。

2. 点对点网络

点对点（P2P）网络是两节点间的通信仅可以采用某一个固定信道进行数据传输的网络。这种网络所采用的通信协议都是基于点对点通信的，如 SLIP（串行线路 Internet 协议）、PPP（点对点协议）、PPPoE（基于以太网的点对点协议）、PPPoA（基于 ATM 的点对点协议）、PPTP（点对点隧道协议）等。

如我们使用的各种 Modem 拨号（拨号网络的基本网络结构如图 2-12 所示）以及路由器间串口（Serial 接口）之间的连接所使用的都是 PPP（点对点协议）或 PPPoE（以太网点对点协议）。我们打电话也是点对点通信的，通信只在两部电话机线路之间进行，其他线路上的用户是听不到的。

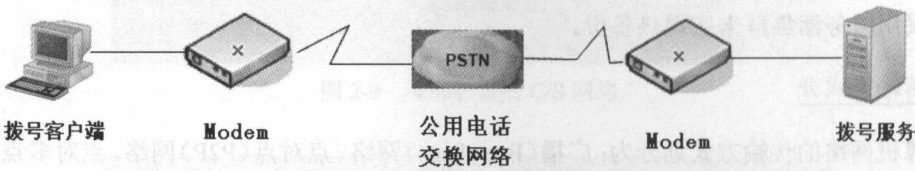


图 2-12　拨号网络基本结构

在点对点传输网络中，数据是以点对点的方式（或者说是“一对一”方式）在计算机或通信设备中传输的，也就是某个端口只能与它相接相连的对端端口进行通信，而不能把数据发送到本网络的其他链路中，也就是只能单点“联系”。因此在点对点传输网络中，不存在信道共享与复用的情况。就像我们现实生活中经常听到传销网络、贩毒集团或者犯罪团伙之间的“单线联系”一样，某个人只能与团伙中的某一个人联系，任何一个下级成员根本不知道整个团伙中其他成员的联系方式。

3. 点对多点网络

点对多点（P2MP）网络不是一种固有的网络类型，而是通过把其他网络类型进行配置改变来实现的。它的网络设备物理连接方式与点对点网络类似，但是它可以通过构建 VC（虚拟通信）的方式实现一个节点可以同时与多个节点（但又不是网络中的所有节点）通信。

如在帧中继网络中可以通过配置把网络类型设置为 P2MP 类型。如图 2-13 所示是一个帧中继网络中三台路由器之间通过 VC 形成的 P2MP 连接。

4. 非广播多路访问网络

非广播多路访问（NBMA）网络的结构与前面介绍的广播网络类型不同，但最关键的是 NBMA 网络不支持广播方式的数据传输（但可以支持组播和单播传播方式）。

在这种网络中，链路层所运行的协议主要是帧中继（FR）、X.25（一种分组交换网络）、ATM（异步传输模式）协议。在 NBMA 网络中，设备之间要进行通信必须先在源设备和目标设备之间建立虚拟通道（VC），在 VC 中又可构建用于不同应用的 VP（虚拟路径）。

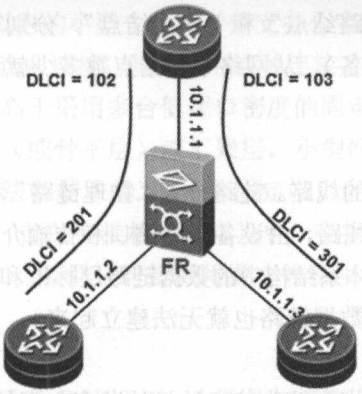


图 2-13 P2MP 网络示例

2.3 计算机网络拓扑结构

拓扑（Topology）学是一种研究与大小、距离无关的几何图形特性的方法。“网络拓扑结构”是由网络节点设备和通信介质通过物理连接所构成的逻辑结构图。网络拓扑结构是从逻辑上表示出网络服务器、工作站的网络部署、相互连接和服务关系。

在不同的计算机网络环境中，可能要采用不同的拓扑结构。在选择拓扑结构时主要考虑的因素有：不同的通信协议类型、不同设备所担当的角色（或者设备间服务的关系）、各节点设备工作性能要求、重新配置的难易程度、维护的相对难易程度、通信介质发生故障时受到影响的设备的情况。

本节将介绍计算机网络（包括局域网和广域网）中的一些主要拓扑结构，在此先介绍与网络拓扑结构有关的几个基本概念。

2.3.1 网络拓扑结构相关基本概念

在设计网络拓扑结构时，我们经常会遇到“节点”“结点”“链路”和“通路”这 4 个术语。它们到底各自代表什么，它们之间又有什么关系呢？

1. 节点

一个“节点”其实就是一个网络端口。节点又分为“转节点”和“访问节点”两类。“转节点”的作用是支持网络的连接，它通过通信线路转接和传递信息，如交换机、网关、路由器、防火墙设备的各个网络端口等；而“访问节点”是信息交换的源点或目标点，通常是用户计算机上的网卡接口。如我们在设计一个网络系统时，通常所说的共有 XX 个节点，其实就是在网络设备上有多个用于连接用户主机、服务器的网络端口。

2. 结点

一个“结点”是指一台网络设备，因为它们通常连接了多个“节点”，所以称为“结点”。在计

算机网络中“结点”又分为“链路结点”和“路由结点”，分别对应的是网络中的交换机和路由器（还可能包括网关或其他三层设备）。从网络中的结点数多少就可以大概知道你的计算机网络规模和基本结构了。

3. 链路

“链路”是两个相邻节点间的线路。链路分为“物理链路”和“逻辑链路”（或称“数据链路”）两种，前者是指实际存在的通信线路，由设备网络端口和传输介质连接实现；后者是指在逻辑上起作用的网络通路，由计算机网络体系结构中的数据链路层标准和对应的数据链路层协议来实现。如果数据链路层协议没有起作用，数据链路也就无法建立起来。

4. 通路

“通路”从发出信息的源节点到接收信息的目的节点之间的一串节点和链路组合。也就是说，它是一系列穿越通信网络而建立起的节点到节点的链路串连。它与“链路”的区别主要在于一条“通路”中可跨越多个结点设备，可包括多条“链路”，而链路仅是指相邻设备之间的那条线路。

2.3.2 星型拓扑结构

星型拓扑结构（Star Topology）也称“集中式拓扑结构”，是因集线器或交换机连接的各节点呈星状（也就是放射状）分布而得名。

1. 基本星型拓扑结构单元

星型拓扑结构是目前应用最广、实用性最好的一种拓扑结构，主要是因为它非常容易实现网络的扩展。无论在局域网中还是在广域网中都可以见到它的身影，但主要应用于有线以太局域网中。但我们这里所谈的“星型拓扑结构”其实是针对一个结构单元（一台集线器或者交换机设备就是一个星型结构单元）而言的，如图 2-14 所示是最简单的单台集线器或交换机星型拓扑结构单元。

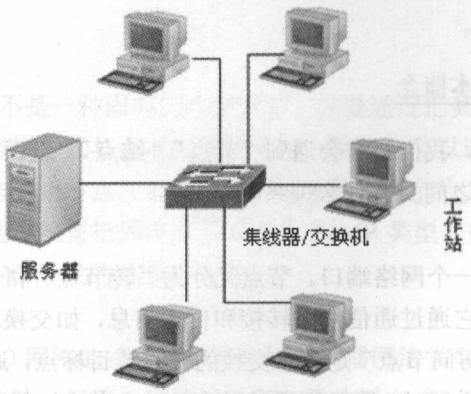


图 2-14 基本星型拓扑结构单元示例

在以上这个星型拓扑结构单元示例中，所有服务器和工作站等网络设备都集中连接在同一台交换机上。因为现在的固定端口交换机最多可以有 48 个交换端口，所以这样一个简单的星型网络完

全可以适用于用户节点数在 40 个以内的小型企业或者分支办公室选用。模块式的交换机端口数可达 100 个以上，可以满足一个小型企业连接。但实际上这种连接方式是比较少见的，因为单独用一台模块式的交换机连接成本还要远高于采用多台低端口密度的固定端口交换机级联方式。模块式交换机通常用于大中型网络的核心层（或骨干层）或汇聚层，小型网络很少使用。

扩展交换端口的另一种有效方法就是堆叠。有一些固定端口配置的交换机支持堆叠技术，通过专用的堆叠电缆连接，所有堆叠在一起的交换机都可作为单一交换机来管理，不仅可以使端口数量得到大幅提高（通常最多堆叠 8 台），而且还可提高堆叠交换机中各端口实际可用的背板带宽，提高了交换机的整体交换性能。

2. 多级星型拓扑结构

在一个以太网中，一般都包括多个集线设备，也就是包括多个星型结构单元。如图 2-15 所示是一个包含两级交换机结构的星型网络，其中的两层交换机通常为不同档次的，可以满足不同需求，核心层（或骨干层）交换机要选择档次较高的，用于连接下级交换机、服务器和高性能需求的工作站用户等，下面各级则可以依次降低要求，以便最大限度地节省投资。

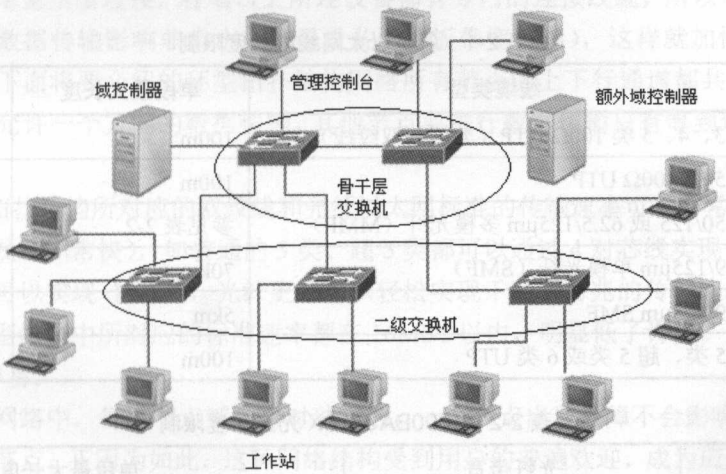


图 2-15 两级星型拓扑结构示例

当然，在实际的大中型企业网络中，其网络结构可能要比图 2-15 复杂许多，还可能有三级甚至四级交换机的级联（通常最多部署四级），还可能有交换机的堆叠和集群。如图 2-16 所示的网络结构中 SS3 Switch 4400 位置就是由两台这样的交换机堆叠组成的。

3. 星型拓扑结构传输距离限制

因为在星型结构网络中通常是采用双绞线或光纤作为传输介质，而单段双绞线的最大长度为 100 米。如果集线设备放置在中心点，这样每一个采用此种结构的集线设备所能连接的网络范围最大直径就达到 200 米，超过这个范围都将采用级联或者中继方法。采用光纤作为传输介质时虽然传输距离可以长许多，但也是有限制的。各种连接线缆长度限制如表 2-1 所示，1000BASE-SX 网络

的光纤长度限制参见表 2-2。

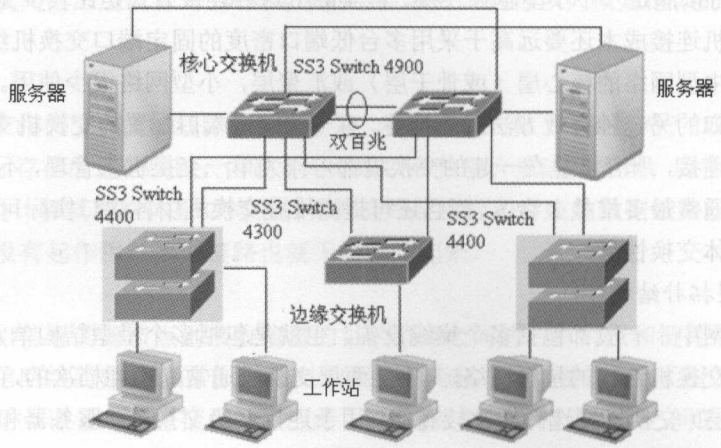


图 2-16　包含交换机堆叠的星型拓扑结构示例

表 2-1　各种以太网线缆长度限制

以太网标准	线缆类型	单段最大长度	网络接口类型
10BASE-T	3、4、5 类 100Ω UTP（非屏蔽双绞线）	100m	RJ-45
100BASE-T	5 类 100Ω UTP	100m	RJ-45
1000BASE-SX	50/125 或 62.5/125μm 多模光纤（MMF）	参见表 2-2	SC 或 ST
1000BASE-LH	9/125μm 单模光纤（SMF）	70km	SC 或 ST
1000BASE-LX	9/125μm SMF	5km	SC 或 ST
1000BASE-T	5 类、超 5 类或 6 类 UTP	100m	RJ-45

表 2-2　1000BASE-SX 光纤长度限制

光纤直径	光纤带宽	单段最大长度
62.5/125μm MMF	160MHz/km	220m
	200MHz/km	275m
50/125μm MMF	400MHz/km	500m
	500MHz/km	550m



许多读者朋友认为，星型拓扑结构只是应用于同楼层的网络中，其实不是这样的。在多楼层，甚至多栋建筑物之间的网络互联多数也是采用星型拓扑结构，因为它具有非常高的传输速率。从表

2-1 和表 2-2 中可以看出，各楼层之间、各建筑物之间都可以采用普通的双绞线（通常是大对数的），但只有距离在双绞线的有效距离范围内。当超过了双绞线的有效距离后，可以采用光纤连接（光纤的传输距离更远，传输性能更好），但光纤介质和相应接口的设备价格更贵，这必须予以充分考虑。同轴电缆只是一种低成本、低性能选择，因为它的传输性能要远低于双绞线和光纤，在目前实际的楼层和建筑物之间的网络互联很少采用。

4. 星型拓扑结构的主要优点

星型拓扑结构的主要优点体现在以下几个方面：

（1）节点扩展、移动方便。

在星型拓扑结构网络中，节点扩展时只需要从交换机等集中设备的空余端口中拉一条电缆连接即可；而要移动一个节点只需要把相应节点设备连接网线从设备端口拔出，然后移到新设备端口即可，并不影响其他任何已有设备的连接和使用，不会像下面将要介绍的环型网络那样“牵一发而动全身”。这是星型拓扑结构的最大优势。

（2）网络传输数据快。

因为整个网络呈星型连接，各端口上所连设备都有专门的连接线缆，所以每个节点的数据传输对其他节点的数据传输影响非常小（仅受设备的背板带宽影响），这样就加快了网络数据传输的速度。不同于下面将要介绍的环型拓扑结构网络所有节点的上下行通道都共享一条传输介质，而且同一时刻只允许一个方向的数据传输，其他节点要进行数据传输只有等到现有数据传输完毕后才可以。

另外，星型拓扑结构所对应的双绞线和光纤以太网标准的传输速率可以非常高（主要是得益于相应的网络技术发展非常快），如普通的 5 类、超 5 类都可以通过 4 对芯线实现 1000Mb/s 传输，7 类屏蔽双绞线则可以实现 10Gb/s，光纤更是可以轻松实现千兆、万兆的传输速率。而后面将要介绍的环型、总线型结构中所对应的标准速率都在 16Mb/s 以内，明显低了许多。

（3）维护容易。

在星型结构网络中，每个节点都是相对独立的，一个节点出现故障不会影响其他节点的连接，可任意拆走故障节点。正因为如此，这种网络结构受到用户的普遍欢迎，成为应用最广的一种拓扑结构类型。但如果集线设备出现了故障，也会导致整个网络的瘫痪。

5. 星型拓扑结构的主要缺点

星型拓扑结构的主要缺点体现在以下两个方面：

（1）核心交换机负荷重。

虽然在星型结构网络中，连接在同一台交换机上的用户间访问可不经过其他设备，但用户要与外部网络或者连接其他设备的用户进行通信，通常都是要通过上层的核心交换机实现的，所以核心交换机的工作负荷比较繁重，容易出现单点故障，要求核心交换机的性能和可靠性都非常高，同时在一些比较大型的网络中可以采用集群的方式来解决单点故障问题。

（2）网络布线较复杂。

在星型结构网络中，每个用户计算机直接采用专门的线缆与集线设备相连，这样整个网络中至

少需要很多布线，使得本身结构就非常复杂的星型网络变得更加复杂了。特别是在大中型企业网络的机房中，太多的线缆无论对维护、管理还是机房安全都是一个威胁。这就要求我们在布线时要多加注意，一定要在各条电缆与集线器和交换机端口上做好相应的标记。同时最建议做好整个布线系统的标记和记录，以备日后出现布线故障时能迅速找到故障发生点。

另外，由于这种星型网络中的每条线缆都是由对应连接端口专用的，利用率不高，在较大的网络中这种浪费还是相当大的。好在像双绞线这种传输介质比较便宜，所以这方面的成本压力其实是非常小的。

2.3.3 环型拓扑结构

环型拓扑结构（Ring Topology）在 20 世纪 90 年代计算机网络刚开始进入国内时比较多采用，应用的标准是 IEEE 802.5。可以说，令牌环在物理上是一个由一系列环接口（称为“中继转发器”，RPU）和这些接口间的点－点链路构成的闭合环路，各站点 PC 通过环接口连到网上。目前这一网络拓扑结构形式已经不用了，因为它的传输速率最高只有 16Mb/s，扩展性能又不好，早已被性能远远超过的星型拓扑结构双绞线以太网替代了。

1. 环型网络结构概述

环型网络拓扑结构主要应用于采用同轴电缆（一般采用成本较低的细同轴电缆）作为传输介质的令牌网中，如图 2-17 所示是一个典型的环型网络。笔者 2000 年曾经到过一个小型公司，为它组建的局域网就是这种结构的。

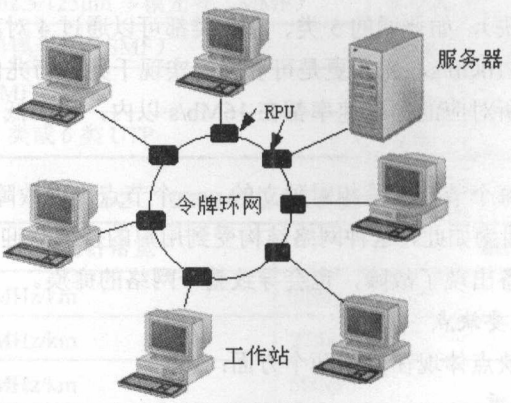


图 2-17　环型拓扑结构网络示例

环型结构网络中的每一站点是通过环中继转发器与它左右相邻的站点串行连接，在传输介质环的两端各加上一个阻抗匹配器（也称“终端匹配器”）就形成了一个封闭的环路，“环型”结构的命名起因就在于此。

在细同轴电缆环型结构网络中的环中继转发器是一个 BNC 接头，阻抗匹配器（链子样的东西）接在 PC 机外壳上（相当于接地），如图 2-18 所示。



图 2-18 BNC 中继转发器和阻抗匹配器

【说明】图 2-17 中只是一种示意图，实际这种拓扑结构的网络不会是所有计算机真的要连接成物理上的环形，可以是任意形状，如直线形、半环形等。这里所说的“环”是从电气性能上来讲的，“环”的形成并不是通过电缆两端的直接连接形成的，而是通过在环的电缆两端加装一个阻抗匹配器来实现的。

环型拓扑结构网络的一个典型代表是采用同轴电缆作为传输介质的 IEEE 802.5 的令牌环网（Token Ring Network）。但令牌环拓扑结构最早是由 IBM 推出的，传输速率为 4Mb/s 或 16Mb/s，较当时只有 2Mb/s 的以太网性能高出好几倍，所以在当时得到了广泛的应用。但随着以太网技术的跳跃式发展，令牌网的技术性能就显得不能适应时代的要求，逐渐被淘汰出局了。其传输原理是，RPU（转发器）从其中的一个环段（称为“上行链路”）上获取帧中的每个比特位信号，然后经过整形和放大转发到另一环段（称为“下行链路”）。如果帧中的目的 MAC 地址与本站点 MAC 地址一致，则复制该 MAC 帧发送给连接本 RPU 的站点。

2. 令牌环网数据传输原理

令牌环网中由点对点链路构成的环路虽然不是真正意义上的广播媒体，但环网上运行的数据帧仍能被所有的站点接收到。而且任何时刻仅允许一个站点发送数据，因此理论上存在发送权竞争问题，实际上就是我们将在第 7 章所讲的“介质争用”问题。

为了解决介质争用问题，在令牌环网中使用了一个称为“令牌”（Token，可以联想起以前武将出征杀敌时手中拿的由皇上发的将牌，代表一种授权）的特殊的 MAC 控制帧（该帧中有一个比特位用来标志令牌的忙或闲状态），使其沿着环路循环。并且规定，只有获得“令牌”的站点才有权发送数据帧（就像战争中只有一位统帅可以获得“将牌”一样），完成数据发送后立即释放令牌以供其他站点使用。由于环路中只有一个令牌，因此任何时刻最多只有一个站点发送数据，不会产生冲突。

如图 2-19 所示是一个简单的令牌环网（注意，这里的环是逻辑意义上的，并不是真正的物理环），其中连接了 4 台机器。现假设 A 站点要向 C 站点发送数据。下面介绍一下 A 站点的具体数据发送流程。

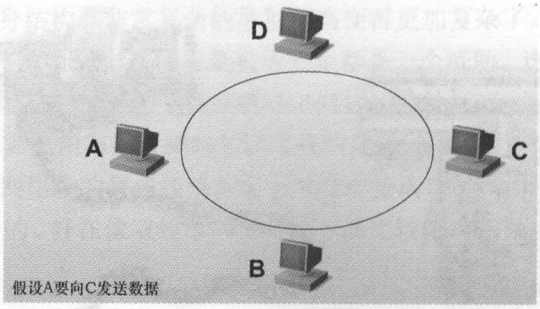


图 2-19　令牌环网示例

（1）环网中的令牌是在网上按照一个方向循环流动的，如图 2-20 所示（图中的“T”代表的就是环网中的“令牌”）。

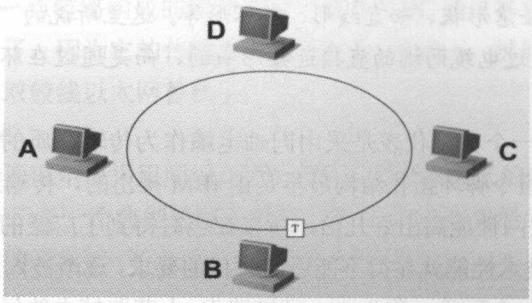


图 2-20　令牌在环网上流动

（2）当令牌转到 A 站点时，A 站点的 RPU 截获该令牌（如图 2-21 所示），把令牌的状态控制位设置为 1，代表处于忙的状态（表示令牌目前已被占用），同时在令牌帧上附加要发送的数据帧沿着环的路径向下发送出去，同时在缓存中保留所发送的数据帧的备份。当令牌帧和数据帧组合（在此称之为“信息帧”）的信息帧流经 B 站点时，B 站点把自己的 MAC 地址与帧中的目的 MAC 地址进行比较，发现不匹配，于是不接收该数据帧，继续转发信息帧，如图 2-22 所示。

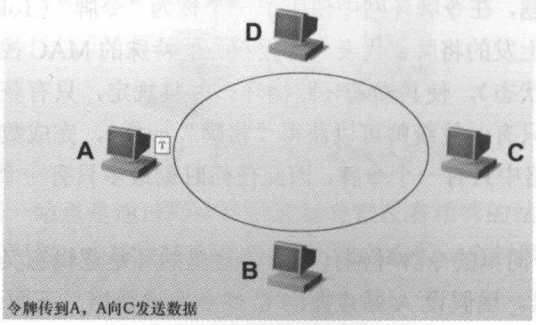


图 2-21　A 站点截获令牌

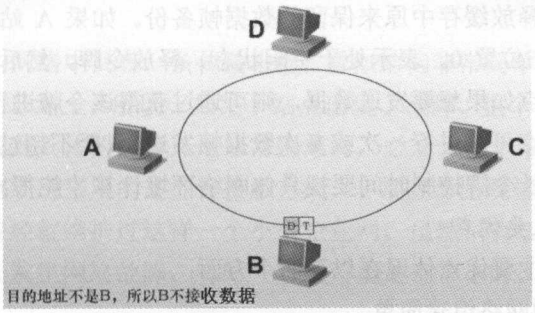


图 2-22　B 站点不接收信息帧

（3）当信息帧流经 C 站点时，C 站点也会把自己的 MAC 地址与帧中的目的 MAC 地址进行比较，发现正好匹配，于是 C 站点的 RPU 复制其中的数据帧并把复制的数据帧传送给 C 站点，丢弃其中的令牌帧，如图 2-23 所示。

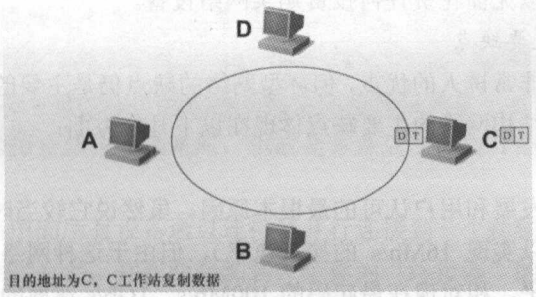


图 2-23　站点 C 的 RPU 复制该数据帧并把其传送给 C 站点

（4）原来的信息帧继续向下传递，流经 D 站点时同样因为它的 MAC 地址与数据帧中的目的 MAC 地址不匹配，因此不接收，继续转，直到回到源站点 A，如图 2-24 所示。

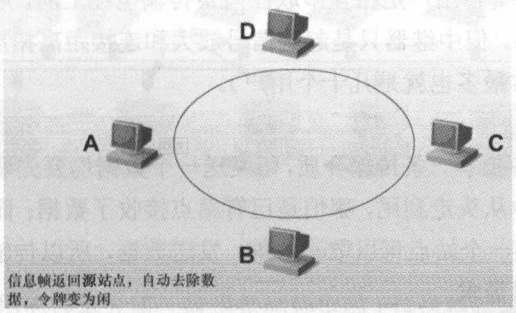


图 2-24　信息帧返回源站点并释放令牌

此时恰好循环一周，A 站点再把原来保存在缓存中的数据帧与返回信息帧中的数据帧进行比较。如发现有错，则从缓存中调用原来的备份数据帧，再复制一份与令牌帧组合在一起重发；如果

经过比较发现没有错，则释放缓存中原来保留的数据帧备份。如果 A 站点没有新的数据要发送，其 RPU 会把令牌帧的状态位置 0，表示处于空闲状态，释放令牌，然后在网上发送出去，令牌继续在环网上流动。其他站点如果想要发送数据，则可通过获得该令牌进行数据发送了。

一个控制了令牌的站点可以进行一次或多次数据帧发送，只要不超过网络中规定的最长令牌控制时间即可。而具体的最长令牌控制时间要视具体网络环境计算才能得出，在此不作具体介绍。

3. 环型拓扑结构的主要优点

环型拓扑结构网络的主要优点体现在以下两个方面：

（1）网络路径选择和网络组建简单。

在这种拓扑结构的网络中，信息在环型网络中流动是一个特定的方向，每两台计算机之间只有一个通路，简化了路径的选择，路径选择效率非常高。同样因为这样，这类网络的组建就相当简单。

（2）投资成本低。

在投资成本低方面，主要体现在令牌环网络中没有任何其他专用网络设备（如交换机），各站点直接通过电缆连接，所以无需花费任何投资购买网络设备。

4. 环型拓扑结构的主要缺点

尽管有以上两个看似非常诱人的优点，但环型网络的缺点仍是主要的，这也是它最终被淘汰出局的根本原因。环型拓扑结构网络的主要缺点体现在以下几个方面：

（1）传输速度慢。

这是它最终不能得到发展和用户认可的最根本原因。虽然说它较当时的 10Mb/s 以太网在速度上有一定优势（因为它可以实现 16Mb/s 的接入速率），但由于这种网络技术后来一直没有任何发展，速度仍保留在原来水平，相对现在最起码的 100Mb/s、1Gb/s 速率的以太网来说，它实在是太落后了。现在连无线局域网（WLAN）的传输速度都远远超过了它。这么低的连接性能决定了它只能承受被淘汰的局面。

（2）连接用户数非常少。

在这种环型拓扑结构中，各用户是相互串联在一条传输电缆上的，所以可以连接的用户数非常受限。尽管可以有中继设备，但中继器只是起到信号放大和连接距离拓展的作用，并不能很明显地提高连接用户的数量（通常最多也就是几十个用户）。

（3）传输效率低。

这种环型拓扑结构网络共享一条传输介质，每发送一个数据均要先取得令牌，每次数据的发送，令牌都要在整个环状网络中从头走到尾，哪怕是已有站点接收了数据；而且每个环型网络中只有一个令牌，所以同一时刻只有一个站点可以取得令牌，发送数据，所以传输效率是非常低的，明显不再适用当前复杂的网络应用需求。

（4）扩展性能差。

因为是环型拓扑结构，而且没有任何可用来扩展连接的设备，决定了它的扩展性能远不如星型拓扑结构的好。如果要新添加或移动站点，就必须中断整个网络，在适当位置切断网线，并在两端作好环中继转发器才能连接。

（5）维护困难。

虽然在环型拓扑结构网络中只有一条传输电缆，看似结构非常简单，但它是一个闭环，设备都连接在同一条串行连接的环路上，所以一旦某个站点出现了故障，整个网络将出现瘫痪。并且在这样一个串行结构中，要找到具体的故障点是非常困难的，必须一个个站点排除，非常不便。

另一方面因为同轴电缆所采用的是插针接触方式，非常容易出现接触不良，造成网络中断，网络故障率非常高。笔者曾经就维护过这样一个小型企业网，虽然只有 20 多台机器，但因为分布在几栋建筑物中，几乎天天发生网络故障，有时一查就可能是几个小时。

2.3.4 总线型拓扑结构

总线型拓扑结构（Bus Topology）与上节介绍的环型拓扑结构从外形上看有些类似，都是共享一条同轴电缆作为传输介质，通过 RPU（中继转发器，或称“连接器”）连接多台计算机，而且网络通信中都是以令牌方式进行的。但还是存在一些区别。

所谓“总线”就是代表网络中连接的各站点间进行数据通信时都必须通过这条传输介质。但总线型拓扑结构采用的是 IEEE 802.4 标准（对应 RFC 1230），接入速率也低于上节介绍的环型网络（只有 10Mb/s）。在当前的局域网中，纯粹的总线型结构网络基本上不见了，取而代之的是在一些特殊的网络环境中与星型拓扑结构混合采用，也就是本章后面即将介绍的混合型拓扑结构。

1. 总线型结构概述

总线型拓扑结构网络中的所有设备通过连接器并行连接到一个传输电缆（通常称为“中继线”“总线”“母线”或“干线”）上，并在两端加装一个称为“终接器”的组件，如图 2-25 所示。终接器主要用于与总线进行阻抗匹配，最大限度地吸收传送端部的能量，避免信号反射回总线产生不必要的干扰。

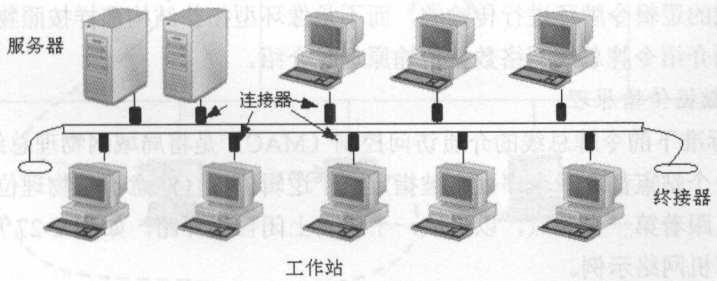


图 2-25 总线型结构示例

总线型结构网络所采用的传输介质一般也是同轴电缆（包括粗同轴电缆和细同轴电缆，也有采用光纤的），如 ATM 网、Cable Modem 所采用的网络等都属于总线型网络结构。为了扩展所连接的计算机数量，可以在网络中添加其他的扩展设备，如中继器。如图 2-26 所示是通过中继器连接的两个总线型网络单元。

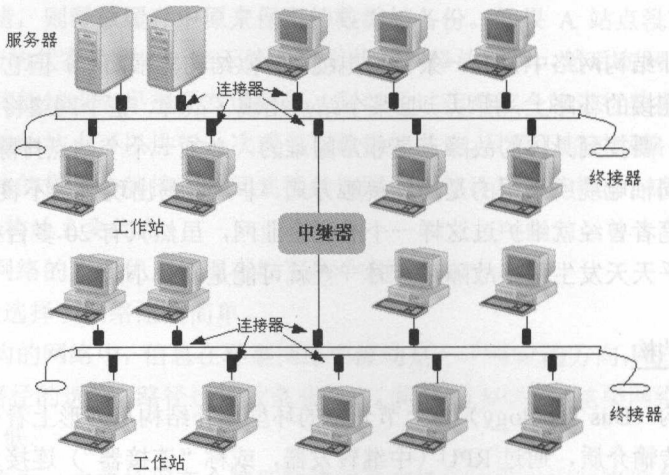


图 2-26　双总线结构网络互联示例

总线型拓扑结构的代表技术就是 IBM 的 ARCNet 令牌网络，所以总线型拓扑结构通常就认为是令牌总线（Token Bus）结构。物理上是总线网，逻辑上是令牌网。

总线型拓扑结构与上节介绍的环型拓扑结构的不同主要体现在以下两个方面：

（1）与传输电缆的连接方式不同。

环型拓扑结构中的连接器与电缆是串联的，所以任何连接站点出现故障都会引起整个网络的中断，而总线型拓扑结构中的连接器与电缆是并联的，所以某个站点发生故障不会影响网络中的其他站点通信。

（2）数据传输原理不同。

两种拓扑结构的数据传输原理不一样，总线型拓扑结构中，令牌帧和数据帧都是沿着根据当前网络环境自动生成的逻辑令牌环进行传输的，而不是像环型拓扑结构那样按照物理环路进行传输。具体将在本节后面介绍令牌总线网络数据传输原理时介绍。

2. 令牌总线数据传输原理

IEEE 802.4 标准下的令牌总线的介质访问控制（MAC）是将局域网物理总线的各站点先生成一个逻辑环，每一个站点都在一个序列中被指定一个逻辑位置（注意不是物理位置），序列中最后一个站点的后面又跟着第一个站点，以形成一个逻辑上闭合的环路。如图 2-27 所示是一个小型总线型拓扑结构计算机网络示例。

从图中可以看出，在物理结构上它是一个总线结构局域网，但是在逻辑结构上又成了一个环型拓扑结构的局域网：A→D→C→G→F→B→E→A。

另外，在总线型拓扑结构网络中，每个站点都知道在它之前的站点和在它之后的邻居站点标识。为了保证逻辑闭合环路的形成，每个站点都动态地维护着一个连接表，该表记录着本站点在环路中的前继、后继和本站点的地址。如图中 A 站点中有 E、D、A 这 3 个地址，分别代表着 A 站点的前继站点（E 站点）MAC 地址、后继站点（D 站点）MAC 地址和本站点 MAC 地址。每个站点根据

它的后继站点 MAC 地址确定下一个可能要占有令牌的站点，所以令牌的传递顺序与总线上各站点的物理位置无关。

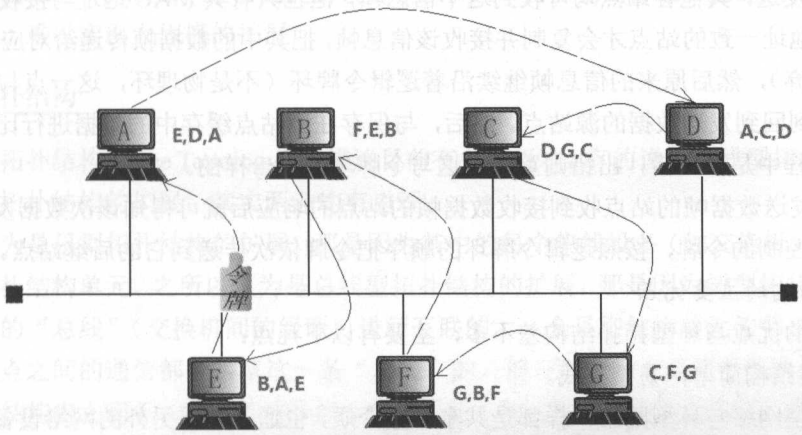


图 2-27　令牌总线中的站点连接表

如果总线网络中某个站点出现故障或者没有工作，则会重新建立新的逻辑令牌网，绕开出现故障和没有工作的站点（因为各站点之间是并行连接的），如图 2-28 所示就是图 2-27 所示总线型拓扑结构网络在出现两个站点（C 和 G 站点）没有正常工作时新建的令牌网。

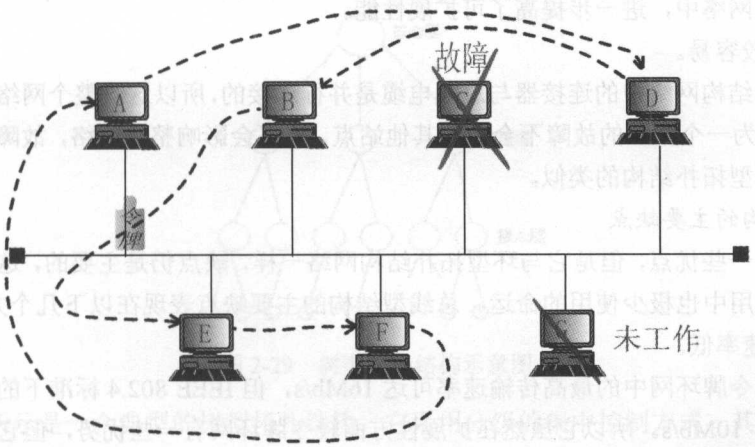


图 2-28　当总线网络中出现不能正常工作的站点时新建的令牌网

令牌总线网络与令牌环网络的数据传输原理基本一样，都是站点在发送数据前必须先取得“令牌”，取得令牌的站点有数据帧要发送则可以发送，如果没有数据帧要发送，则直接把令牌传递到逻辑令牌环中的后继站点。而且只有获得令牌的站点才能发送信息，其他站点只能接收信息或者被动地发送信息（在拥有令牌的站点要求下发送信息）。由于站点接收到令牌的过程是顺序依次进行

的，因此对所有站点都有公平的访问权，也不会出现介质访问冲突。

在具体的数据发送过程中，取得了令牌的站点把令牌帧和数据帧一起（在此也称为“信息帧”）在总线网络中发送，其他各站点均可收到这个信息帧，但也只有其 MAC 地址与接收到的数据帧中的目的 MAC 地址一致的站点才会复制并接收该信息帧，把其中的数据帧传递给对应的站点 PC（其中的令牌帧丢弃），然后原来的信息帧继续沿着逻辑令牌环（不是物理环，这一点与令牌环网不一样）传递，直到回到发送数据的源站点。然后，与保存在源站点缓存中的数据进行比较，用于校验数据在传输过程中是否出错，出错则重传，这与令牌环网是一样的。

另外，当发送数据帧的站点收到接收数据帧的站点的响应后就可得知该次数据发送成功，随即释放自己所有控制的令牌，按照逻辑令牌环的顺序把令牌依次传递到它的后继站点。

3. 总线结构的主要优点

总线拓扑的优点与环型拓扑结构差不多，主要有以下几点：

（1）网络结构简单，易于布线。

因为总线型网络与环型网络一样都是共享传输介质，也通常无需另外的网络设备，所以整个网络结构比较简单，布线比较容易。

（2）扩展较容易。

这是它相对同样是采用同轴电缆（或光纤）作为传输介质的环型网络结构的最大的一个优点。因为总线型结构网络中，各站点与总线的连接是通过连接器并行连接的（环型网络中连接器与电缆的连接是串行的），所以站点的扩展无需断开网络，扩展容易了许多。而且还可以通过中继器设备扩展连接到其他网络中，进一步提高了可扩展性能。

（3）维护较容易。

因为总线型结构网络中的连接器与总线电缆是并行连接的，所以这给整个网络的维护带来了比较大的便利，因为一个站点的故障不会影响其他站点，更不会影响整个网络，故障点的查找就容易了许多。这与星型拓扑结构的类似。

4. 总线结构的主要缺点

尽管有以上一些优点，但是它与环型拓扑结构网络一样，缺点仍是主要的，这些缺点也决定了它在当前网络应用中也极少使用的命运。总线型结构的主要缺点表现在以下几个方面：

（1）传输速率低。

IEEE 802.5 令牌环网中的最高传输速率可达 16Mb/s，但 IEEE 802.4 标准下的令牌总线标准最高传输速率仅为 10Mb/s。所以它虽然在扩展性方面较令牌环网有一些优势，但它同样摆脱不了被淘汰的命运。

（2）故障诊断困难。

虽然总线拓扑结构简单、可靠性高，而且是互不影响的并行连接，但故障的检测仍然很不容易。这是因为这种网络不是集中式连接，故障诊断需要在网络中的各站点计算机上分别进行。另外，在这种结构中如果故障发生在各个计算机内部，只需要将计算机从总线上去掉，比较容易实现。但是如果是总线传输介质发生故障，则需要全部更换相应段的传输介质。

（3）难以实现大规模扩展。

虽然相对环型网络来说，总线型网络结构在扩展性方面有了一定的改善，可以在不断开网络的情况下添加设备，还可以添加中继器之类的设备予以扩展，但仍受到传输性能的限制，其扩展性远不如星型网络，难以实现大规模的扩展。

2.3.5 树型拓扑结构

关于树型拓扑结构（Tree Topology）的描述目前有多种版本，有的说是总线型拓扑结构的扩展，有的说是星型拓扑结构的扩展，其实两者均有道理。

之所以认为是星型拓扑结构的扩展，那是因为其中的每个集线设备（如交换机）所连接的就是一个个星型拓扑结构单元。之所以认为是总线型拓扑结构的扩展，那是因为树型拓扑结构中各设备间是通过类似的“总线”（交换机间的级联）进行互联的，一个星型结构单元的节点与另一个星型结构单元的节点之间的通信都是共享这一条“总线”的。但一般认为它是星型拓扑结构的扩展。

树型拓扑结构自上而下（从核心交换机（或骨干层）到汇聚层，再到边缘层）是依次分层扩展的，就像一棵倒放的“树”，这或许就是把它定义为“树型”拓扑结构的原因之一。树型拓扑结构的最顶层（也就是核心层）相当于树的“根”，中间层（汇聚层）相对于“树的枝”，而最下层（边缘层或接入层）则相当于树枝上的“细枝”和“树叶”。自上而下，所用的交换机数量逐层增多，简化的树型拓扑结构如图 2-29 所示（图中每一个大圆圈代表一台交换机，最下面的每个小圆圈代表一台所连接的主机）。

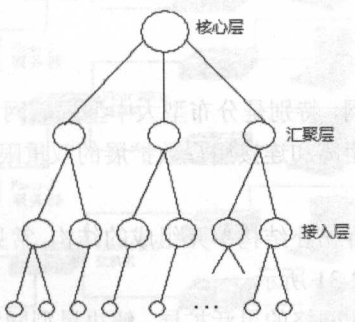


图 2-29 树型拓扑结构示意图

如图 2-30 所示是一个典型的树型拓扑结构。它采用分级的集中控制方式，其传输介质可以有多条分支，但不形成闭合回路，每条通信线路都必须支持双向传输。

从以上介绍可以得知，树型拓扑结构中包含了一个个星型结构单元，所以树型拓扑结构的主要优点和缺点与星型结构是对应一样的。其中一个最主要的优点就是扩展方便。星型拓扑结构便于扩展我们已经知道，只要在集线设备空余的端口上拉出一条网线就可以添加新的设备；而在树型拓扑结构中，通过多级星型结构的级联可以更方便地实现在连接距离和端口数据上的扩展，实现更大规模网络的扩展升级。

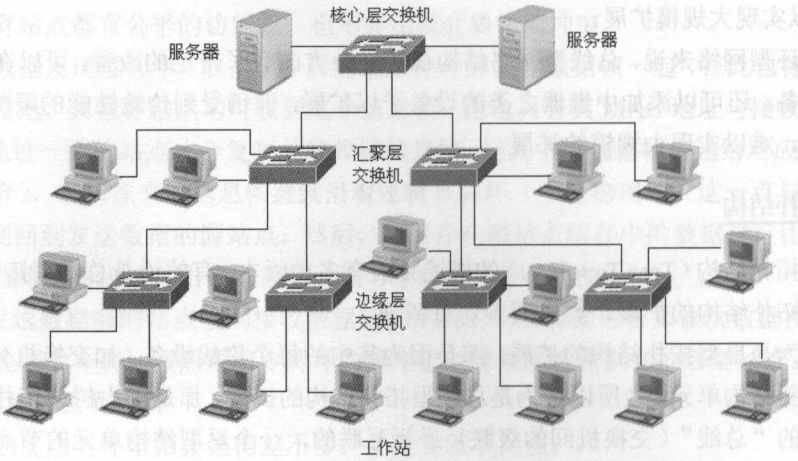


图 2-30　典型树型拓扑结构示例

另外，树型拓扑结构也像星型拓扑结构一样便于维护，因为每个用户计算机都用单独的线缆与交换机连接，任何用户出现了故障都不影响其他用户计算机与网络的连接。当然交换机间的连接如果出现了故障，还是会影响部分甚至整个网络的通信，如核心层交换机连接外部网络的线路出现了故障，则内部网络的所有用户均不能与外部网络通信。

至于树型拓扑结构的主要缺点也与星型拓扑结构一样，高层交换机（如核心层交换机、汇聚层交换机）由于工作负荷大，容易出现单点故障，具体参见前面星型拓扑结构的介绍。

2.3.6　混合型拓扑结构

混合型网络结构是目前局域网，特别是分布型大中型局域网中应用最广泛的网络拓扑结构。它可以解决单一网络拓扑结构传输距离和连接用户数扩展的双重限制。

1. 混合型结构概述

混合型网络拓扑结构是指多种拓扑结构单元组成的结构，常见的是由星型拓扑结构和总线型拓扑结构结合在一起组成的，如图 2-31 所示。

混合型拓扑结构更能满足较大网络的灵活扩展，解决星型网络在传输距离上的局限（因为双绞线的单段最大长度要远小于同轴电缆和光纤），同时又解决了总线型网络在连接用户数量上的限制。图 2-31 所示只是一种简单的混合型网络结构，实际上的混合型拓扑结构主要应用于分布在多层或者多栋建筑物中的网络。其中采用同轴电缆或光纤的“总线”用于垂直或横向干线，基本上不连接工作站，只是连接各楼层或各建筑物中的各核心层交换机，而其中的星型拓扑结构网络则体现在各楼层或者各建筑物内部的各用户网络中，如图 2-32 所示。

在实际的组网中，现在一般很少使用同轴电缆作为总线了，而是采用传输性能更好更方便进行网络连接的大对数双绞线，因为在一般的 20 层以内的楼中 100 米的双绞线就可以满足，如图 2-33 所示。

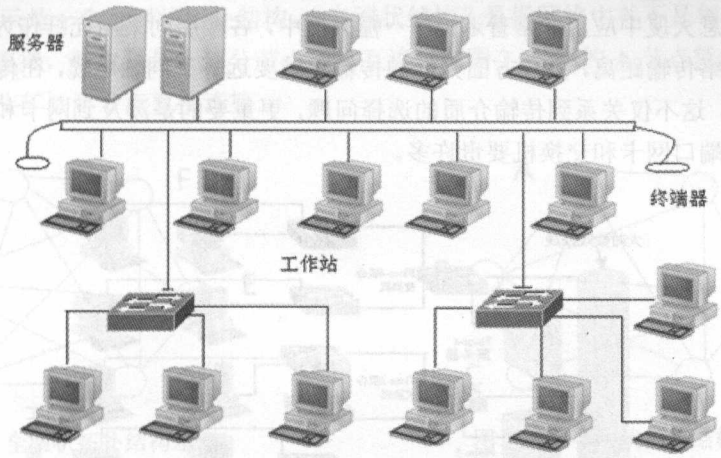


图 2-31　混合型拓扑结构示例

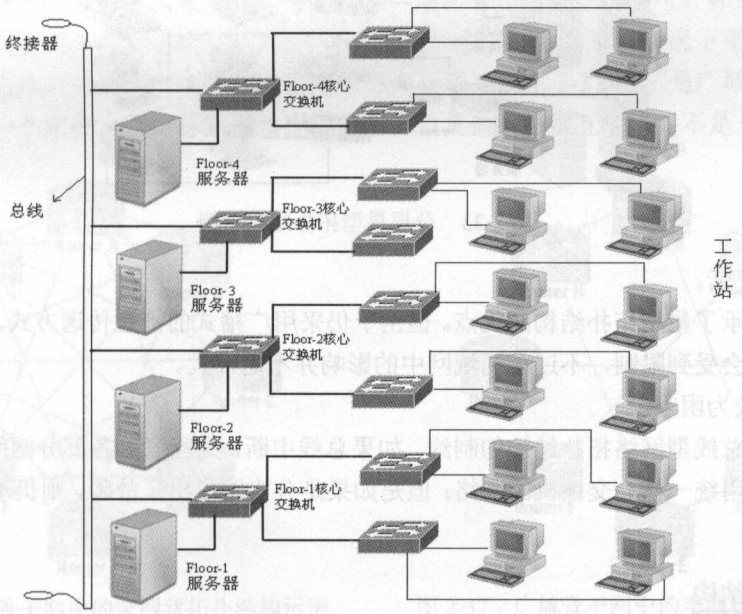


图 2-32　多楼层的混合型网络结构示例

如果距离过远，如高楼层或者多个建筑物之间的网络互联，则可以用光纤作为传输介质，无论哪一种，传输性能均要比总线型连接方式好许多。

2. 混合型拓扑结构的主要特点

混合型拓扑结构主要有以下几方面的特点：

（1）应用广泛。

这主要是因为它解决了星型和总线型拓扑结构的不足，满足了大公司组网的实际需求。目前在

一些智能化的信息大厦中应用非常普遍。在一幢大厦中，各楼层间采用光纤作为总线传输介质，一方面可以保证网络传输距离，另一方面光纤的传输性能要远好于同轴电缆，在传输性能上也给予了充分保证。当然，这不仅关系到传输介质的选择问题，更重要的是涉及到网卡和交换机端口类型的选择，光纤类型端口网卡和交换机要贵许多。

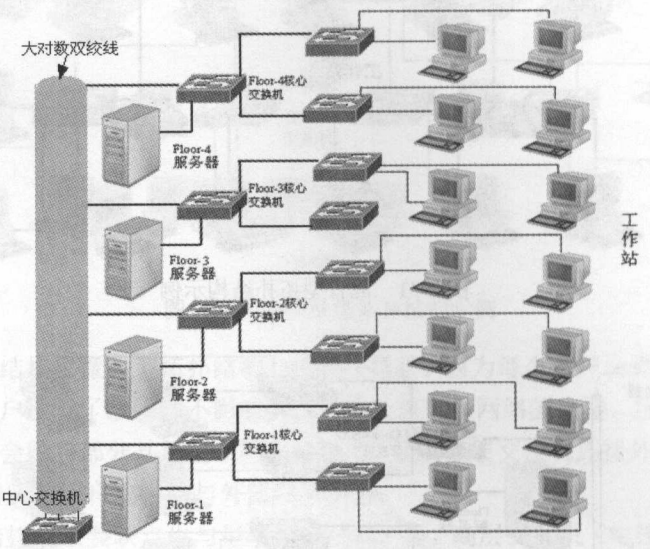


图 2-33　分层星型拓扑结构示例

（2）扩展灵活。

这主要是继承了星型拓扑结构的优点。但由于仍采用广播式的消息传送方式，所以在总线长度和节点数量上也会受到限制，不过在局域网中的影响并不是很大。

（3）维护较为困难。

这主要受到总线型网络拓扑结构的制约，如果总线中断，则互联的各部分网络也就中断了，特别是对于那些使用统一核心交换机的网络；但是如果是分支网段出了故障，则仍不影响整个网络的正常运作。

2.3.7　网状拓扑结构

网状拓扑结构（Mesh Topology）又称为“无规则型”拓扑结构。在这种结构中，各节点之间通过传输介质彼此互联，构成一个类似“鱼网”一样的结构，因此而得名。

这种拓扑结构与前面介绍的几种拓扑结构不一样，因为它的最高追求是可靠性（是所有拓扑结构中最高的），同时成本会大大增加（需要更多的网线，端口利用率也会有较大降低），所以主要应用于广域网中（前面几种拓扑结构主要应用于局域网中）。

网状拓扑结构又有“全网状结构”和“半网状结构”两种。“全网状结构”是指网络中任何两个节点间都是相互连接的。假设一个网络中有 n 个节点，则任何一个节点就有 n-1 条与其他节点的

连接。如图 2-34 所示是一个全网状拓扑结构。“半网状结构”是指网络中并不是每个节点都与网络的其他所有节点有连接，可能只是一部分节点间有互连，如图 2-35 中的 A 节点就没有与 E 节点直接连接，C 节点也没有与 F 节点直接连接。

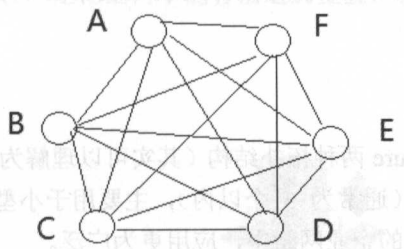


图 2-34　全网状拓扑结构示例

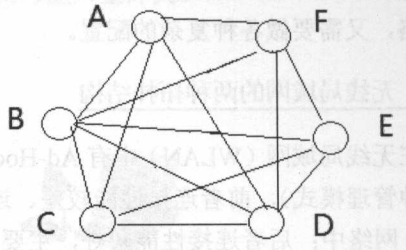


图 2-35　半网状拓扑结构示例

广域网中采用网状拓扑结构的主要目的就是实现链路或路由线路冗余，提高网络的可靠性。当然一般并不会在整个广域网中都采用这种拓扑结构，而主要是在骨干网络中。如图 2-36 所示是一个广域骨干网全网状拓扑结构示例，其中各个路由器之间彼此互连。更多情况下采用的是半网状拓扑结构，即仅少数结点需要与网络中的其他所有节点互连。如图 2-37 所示是广域骨干网中采用半网状拓扑结构的一个示例，其中各路由器也只是与少数其他路由器互连，并不是全部互连。

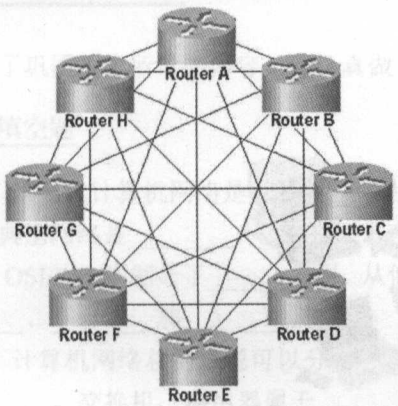


图 2-36　广域骨干网中的全网状拓扑结构示例

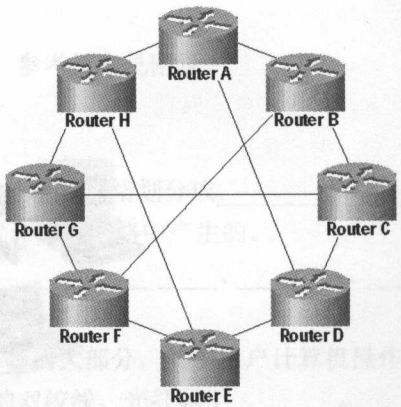


图 2-37　广域骨干网中的半网状拓扑结构示例

网状拓扑结构的主要优点就是具有极高的可靠性，因为这种拓扑结构中任何节点的通信都存在多条路径，使得任何单一，甚至多条连接线路中断都不会影响用户数据从一个节点到网络中另一个节点的转发。这对于 ISP 的骨干网络是非常需要的，因为这样的网络可靠性才是最重要的。

网状拓扑结构的优点同时也带来了它自身的一个最大缺点，那就是成本高。因为不在广域网中，各设备可能分布在不同地点（甚至不同城市、不同国家），要实现全部互联可不是件简单的事，远不是局域网中多拉根网线那么简单的事了。企业的广域网中连接通常是租用 ISP 拨号网络或专线，但对于 ISP 自身来说，肯定是要自己拉光纤来连接不同地点的设备了，这样的投入肯定不是一般人

可以想象得到的。

另一方面，每台设备之间都要互联，结构非常复杂，不仅配置会很复杂，也不易于网络的维护与管理，而且不便于网络扩展，因为任何新加的一台设备都要与其他所有原设备进行连接，还要在每台设备上进行相关的配置。另外，由于节点间存在多条冗余线路，导致容易出现路由环路或者二层环路，又需要做各种复杂的配置。

2.3.8 无线局域网的两种拓扑结构

在无线局域网（WLAN）中有 Ad-Hoc 和 Infrastructure 两种拓扑结构（其实可以理解为 WLAN 的两种管理模式），前者连接性能较差、连接用户较少（通常为 5 个以内），主要用于小型家庭、SOHO 网络中；后者连接性能较好，主要用于较多用户的企业网络中，应用更为广泛。

1. 无 AP 的 Ad-Hoc 模式的主要优缺点

Ad-Hoc 对等 WLAN 模式采用的是点对点连接方式（如图 2-38 所示），只能单点通信，就像有线网络中的对等网一样，所以连接性能较差，仅适用于较少数量的计算机无线互联（通常是在 5 台主机以内）。同时由于这一模式没有中心管理单元，所以这种网络在可管理性和扩展性方面受到一定的限制。而且各无线节点之间只能单点通信，不能实现交换连接。当然这一无线网络结构还是有它的自身优点的，那就是网络结构简单，只要安装了无线网卡即可连接，无需其他网络设备，成本非常低。

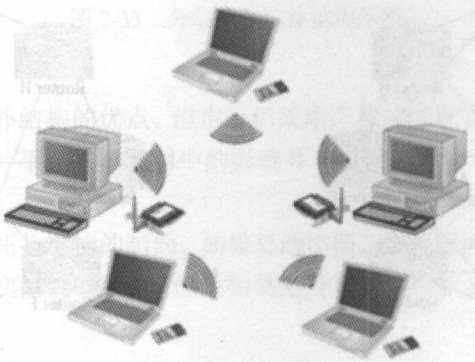


图 2-38 Ad-Hoc 对等无线局域网结构

2. 基于 AP 的 Infrastructure 结构

这种基于无线 AP 的 Infrastructure 基础结构模式其实与有线网络中的星型交换模式差不多（如图 2-39 所示），除了各无线用户需要安装无线网卡外，还需要一个用于集中连接各无线用户的无线 AP，它相当于有线网络中的集线器。无线 AP 都提供了一个有线以太网接口，用于与有线网络、工作站和路由设备的连接。这种网络结构模式主要表现在网络易于扩展、便于集中管理、能提供用户身份验证等优势，另外数据传输性能也明显高于 Ad-Hoc 对等结构。

其实图 2-39 所示也只是一个 Infrastructure 基础结构单元，在实际的企业 WLAN 网络中，可能

有许多台 AP 设备，而且还可能有 WLAN 天线、WLAN 中继器、WLAN 网桥、WLAN 控制器等设备。WLAN 天线和 WLAN 中继器可用于连接更远距离的 WLAN 用户，WLAN 网桥可用于连接不同的 WLAN 网段，WLAN 控制器则可对整个 WLAN 网络进行管理。这就涉及到信道的分配和优化了，在一定范围内不能有信道的重叠，否则信号之间就可能产生冲突。

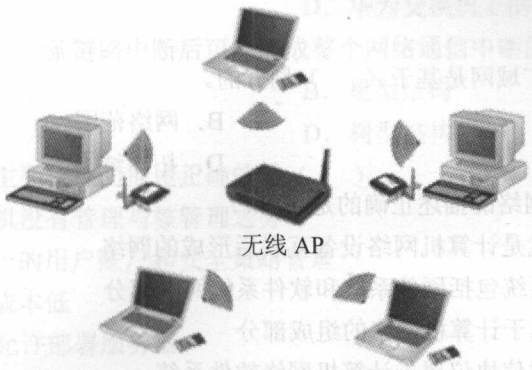


图 2-39 Infrastructure 基础无线局域网结构

2.4 课后自测题

为了巩固本章所学的内容，请认真做下面的自测题，参考答案见附录。

2.4.1 填空题

1．第一代计算机网络是以_____为中心，第二代计算机网络划分成_____和_____两部分，典型网络是_____，OSI/RM 是在第_____代计算机网络中产生的。

2．OSI/RM 共划分了_____层，从低到高依次是_____、_____、_____、_____、_____、_____和_____。

3．计算机网络总的来说可以分为_____和_____两大部分，其中用户计算机操作系统属于_____，交换机、路由器属于_____，连接网络的双绞线、光纤属于_____。

4．计算机网络按照所覆盖的范围可分为_____、_____和_____3 种。我们平时用的银行柜员机所在的网络属于_____，WLAN 无线网络属于_____，总公司与分支公司位于不同的城市，通过 VPN（虚拟专用网）连接起来的网络属于_____。

5．按管理模式可将计算机网络划分为_____和_____两种，我们经常用的电子邮件服务、DNS 服务属于_____模式。

6．按计算机网络传输方式划分为_____、_____、_____和_____4 种，其中以太网属于_____，Modem 拨号网络属于_____，X.25、ATM 网络属于_____。

7．在计算机局域网中常见的拓扑结构包括_____、_____、_____和_____4 种，

其中在当前企业网络中最广泛使用的是_____。在计算机广域网中最常采用的拓扑结构是_____，在局域网和广域网中都常用的拓扑结构是_____。

8．以太网使用的标准是_____，令牌环网使用的标准是_____，令牌总线网采用的标准是_____。

2.4.2 选择题（可多选）

1．计算机局域网与广域网是基于（　）划分的。

A．通信协议　　B．网络范围

C．传输介质　　D．拓扑结构

2．下面关于计算机网络的描述正确的是（　）

A．计算机网络就是计算机网络设备互联后形成的网络

B．计算机网络系统包括硬件系统和软件系统两大部分

C．传输介质不属于计算机网络的组成部分

D．计算机网络通信协议属于计算机网络软件系统

3．最早的计算机网络通常认为是（　）。

A．ARPANET　　B．BITENT

C．CERNET　　D．ETHERNET

4．以下关于第一代计算机网络的描述不正确的是（　）。

A．以计算机主机为中心

B．易出现单点故障

C．用户使用的是不带数据处理能力的终端设备

D．不能使用集中器等设备

5．以下关于第二代计算机网络的描述不正确的是（　）。

A．典型代表是 APPANET　　B．以计算机主机为中心的联机系统

C．使用 IMP 专门负责通信处理　　D．首次使用“存储－转发”数据交换方式

6．在神舟七号宇宙飞船飞向太空时，负责控制飞船运行的计算机网络中有些任务是需要位于不同节点的多个操作员协同完成的。这体现了计算机网络的（　）功能。

A．资源共享　　B．协同工作

C．网络互联　　D．数据传输

7．小王出差到了北京，打电话给广州公司的小张说他要公司的一份文件，小张马上用邮件发给他。这体现了计算机网络的（　）功能。

A．资源共享　　B．协同工作

C．网络互联　　D．数据传输

8．在一个公司网络中，小张给管理员打电话说要看一份公司的管理文件，管理员说在文件服务器上有，自己看吧。这体现了计算机网络的（　）功能。

A．资源共享　　B．协同工作

C．网络互联　　D．数据传输

9．下列不属于计算机网络组成部分的是（　　）

A．计算机 BIOS 系统　　B．计算机主机

C．Modem　　D．华为交换机上的 VRP 系统

10．下列拓扑结构中，一条链路中断后可能造成整个网络通信中断的是（　　）。

A．环型结构　　B．星型结构

C．总线型结构　　D．树型结构

11．下列关于对等网主要特点的描述正确的是（　　）。

A．网络中的主机没有管理与被管理之分

B．无法进行统一的用户账户和安全策略管理

C．配置容易，成本低

D．对等网中不允许部署服务器

12．下面关于拓扑结构的描述正确的是（　　）。

A．网络上每个节点发生故障时就可能使整个网络瘫痪的拓扑结构是星型结构

B．每一种网络只允许包含一种拓扑结构

C．环型结构比其他结构的成本更高

D．拓扑结构描述的是各节点间通信的路径

13．下列拓扑结构中可靠性最高的是（　　）。

A．星型结构　　B．网状结构

C．树型结构　　D．总线型结构

14．下面有关令牌环网络的描述错误的是（　　）。

A．同一时间令牌只能沿着逻辑环的一个方向进行传递

B．任何时刻只有一个站点发送数据，不会产生介质访问冲突

C．发送数据时令牌必须随着数据一起传递，并一起被目的站点接收

D．发送的信息帧最终都会回到源站点，目的站点接收的是复制数据帧

15．下面有关令牌总线网络的描述错误的是（　　）。

A．数据传输也是沿着逻辑环进行的，但每次生成的逻辑环可能都不一样

B．逻辑环中每个站点都有一张“连接表”，标识了本站点及前后站点的地址

C．如果总线网络中的某个站点出现故障或者没有工作，可以重新建立新的逻辑环

D．发送数据时不同时发送令牌帧，源数据也不用回到源站点

16．下面关于令牌环网络和令牌总线网络的比较不正确的是（　　）。

A．站点发送数据前必须拥有“令牌”，发送的源数据必须回到源站点

B．都是沿着逻辑环进行传递的，而且逻辑环都可以根据实际网络结构重新生成

C．都是通过比较数据帧中的 MAC 地址与站点的 MAC 地址来确定是否接收数据的

D．网络中的某台机器出现了故障都不会影响数据传递

17．以下是混合型结构特点的是（　　）。

A．易扩展　　B．易出现单点故障

C．不易管理　　D．成本高

18．以下不是网状结构特点的是（　　）。

A．可靠性强　　B．成本高

C．易实现负载均衡　　D．配置容易

19．在家中把两台笔记本电脑直接通过 WLAN 实现互联所采用的拓扑结构是（　　）。

A．Ad-Hoc 结构　　B．星型结构

C．点对点结构　　D．Infrastructure 结构

3

计算机网络体系结构

从本章起就开始正式介绍计算机网络体系结构（Computer Network Architecture）了，本章先从宏观角度介绍各种计算机网络体系结构。本章的内容虽然看似简单，但确实对于理解计算机网络结构的划分、各层的主要用途，以及计算机网络通信原理非常重要。

本章首先介绍的是计算机网络体系结构的基本发展历程，介绍了已出现的几种典型的计算机网络结构；然后着重以类比的方式通俗化剖析了OSI/RM和TCP/IP体系结构在层次划分、各层主要用途设计方面的考虑，使大家可以更加轻松地理解计算机网络体系结构层次划分的必要性和各层的主要用途；接着着重介绍了计算机网络的基本通信原理，其中包括数据在发送端和接收端各自的传输方向、体系结构中各层的数据传输单元、各层的数据封装/解封装原理；最后介绍的是计算机网络通信协议方面的基本组成。本书后面的各章将具体介绍这些体系结构中各层主要功能的实现原理、主要通信协议，以及相关的计算机网络基础知识。

本章重点与难点：

- 计算机网络体系结构的设计和层次划分考虑
- 计算机网络体系结构各层的主要用途
- 计算机网络通信原理（包括通信流程、对等会话原理、协议封装和解封装）
- 计算机网络通信协议的三要素

3.1 计算机网络体系结构发展历程

计算机网络体系结构从宏观、逻辑意义上描述整个计算机网络通信功能划分的分层结构，也是

由计算机网络软/硬件系统共同实现的。计算机网络体系结构与第 2 章所介绍的计算机网络一样也有一个渐进式的发展过程，一开始并不是像我们现在所提到的 OSI/RM 或 RCP/IP 体系结构那样，而是经历了一个如图 3-1 所示的基本历程。本节就先来为大家理一下计算机网络体系结构的基本发展历程。

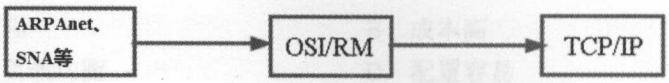


图 3-1　计算机网络体系结构基本发展历程

1. ARPAnet 体系结构

计算机网络体系结构自然是先有了真正意义的计算机网络后才出现的。在第 2 章介绍计算机网络的发展历史中就已知道 DARPA（Defence Advanced Research Projects Agency，美国国防部高级研究计划署）首次把它开发的 ARPAnet 网络划分成了“资源子网”和“通信子网”两个层次，如图 3-2 所示。这是计算机网络体系结构的雏形，后面许多公司提出的新的计算机网络体系结构都是这种基础划分进行细化的。

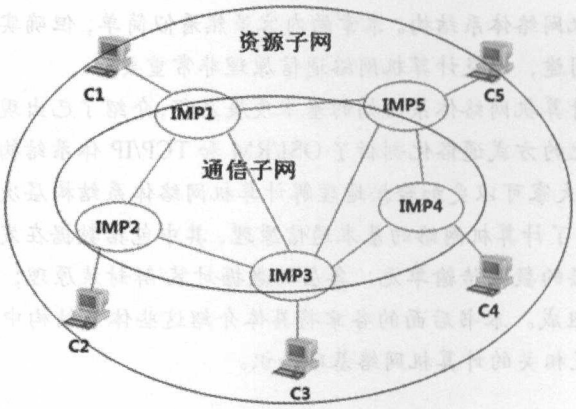


图 3-2　ARPAnet 的两层体系结构

2. SNA 体系结构

美国 ARPAnet 中的“资源子网”和“通信子网”两层结构划分方式显然是比较粗的，还不足以真正体现计算机网络通信的原理。于是后面又有许多大公司和科研机构对计算机网络体系结构进行了研究，其中最具有代表性的就是 IBM 公司提出的 SNA 体系结构。

在第二代计算机网络时期，1974 年美国的 IBM 公司提出了基于其大型机的“主机－终端”通信模型——SNA（Systems Network Architecture，系统网络体系结构）。SNA 把整个计算机网络分成 7 个层次，如图 3-3 所示。各层的简单介绍可在 IBM 公司网站的这个页面进行了解（大家会发现它与后面将要介绍的国际标准化的 OSI/RM 体系结构非常类似）：http://docwiki.cisco.com/wiki/IBM_Systems_Network_Architecture_Protocols。

3. OSI/RM 体系结构

在 IBM SNA 体系结构产生后，许多其他大型公司也纷纷推出自己的网络体系结构，如 1975 年 DEC 推出了 DNA（Digital Network Architecture，数字网络体系结构），但它们都是各自为政，彼此不兼容。

为了解决这一问题，ISO 成立了专门的研究机构，于 1983 年最终在原来 IBM SNA 结构的基础上，以国际标准形式发布了第一个国际标准化的 OSI/RM（Open Systems Interconnection Reference Model）体系结构，如图 3-4 所示。由此就进入了第三代计算机网络时期。

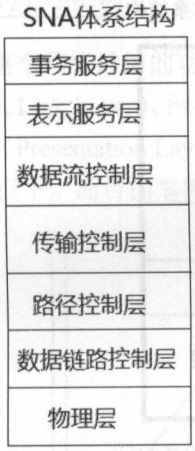


图 3-3 IBM SNA 体系结构

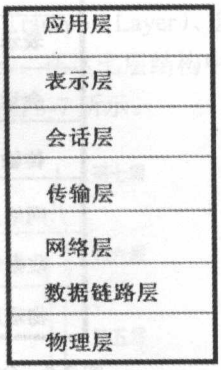


图 3-4 OSI/RM 体系结构

对比图 3-3 所示的 SNA 体系结构就可以看出，OSI/RM 与 SNA 体系结构是相当类似的，都是七层结构，而且尽管有些层次的名称不一样，但实际功能却基本一样。

另外，OSI/RM 体系结构也继承了计算机网络体系结构雏形——ARPAnet 在体系结构划分上的设计思想，因为 ISO 把 OSI/RM 的 7 个层次也分成了“资源子网”和“通信子网”两部分。其中“资源子网”包括高四层，即“传输层”“会话层”“表示层”和“应用层”，而“通信子网”包括低三层，即“物理层”“数据链路层”和“网络层”。

因为 OSI/RM 是由国际标准化组织 ISO 提出的，是通用的，这样就为各计算机网络软/硬件开发厂商提供了统一的标准，解决了不同厂商产品的不兼容问题，极大地促进了计算机网络技术、产品和应用的发展。也正因为如此，后面很快又诞生了一系列 IEEE 计算机网络标准。

4. TCP/IP 体系结构

理论上来讲，有了国际标准化的 OSI/RM 后，在计算机网络体系结构上应该不会有人另起炉灶，开发新的计算机网络体系结构了。但事实就这么具有戏剧性，也就是现在大家都知道的，事实上我们今天的计算机网络已不再是直接采用 OSI/RM 这个国际标准化体系结构，而是采用一种称为 TCP/IP 的非国际化标准。

之所以会出现这样戏剧性的结果，是因为 OSI/RM 这个体系结构自身仍存在比较明显的不足，主要是在高三层中“会话层”和“表示层”功能太弱，区分不明显，这也给计算机网络软/硬件产品的开发带来许多不便。于是 TCP/IP 体系结构就把 OSI/RM 体系结构中的最高三层合并成了一层——应用层。

另外，在最初的 TCP/IP 体系结构中还把 OSI/RM 中的最低两层（“物理层”和“数据链路层”）合并成了“网络接口层”这一层。这样一来，TCP/IP 体系结构只有四层了。图 3-5 所示是原来的 OSI/RM 体系结构与 TCP/IP 体系结构的对比。

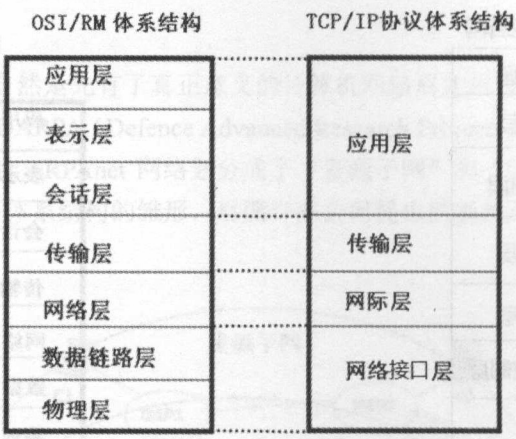


图 3-5　OSI/RM 体系结构与 TCP/IP 体系结构的对比

后来发现把 OSI/RM 最低两层合作的做法并不好，于是当前实际应用的 TCP/IP 体系结构是五层，把原来最低的“网络接口层”仍与 OSI/RM 一样划分成了两层，即“数据链路层”和“物理层”。图 3-6 所示是四层和五层 TCP/IP 体系结构的对比。

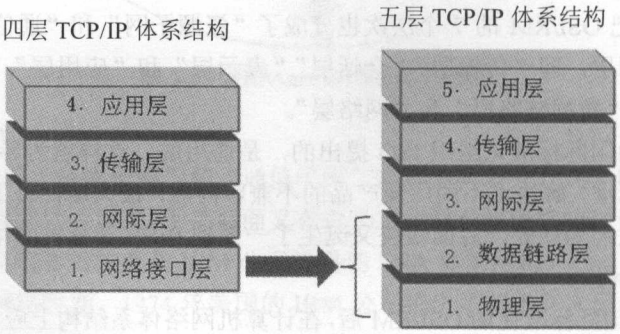


图 3-6　四层和五层 TCP/IP 体系结构的对比

以上是从发展历程角度对几种典型的计算机网络体系结构进行的简单介绍，下面对当前主要的 OSI/RM 和 TCP/IP 体系结构进行具体介绍。因为 TCP/IP 体系结构其实主要部分与 OSI/RM 是一样

的，所以本章后面所介绍的各种计算机网络通信原理方面是同时适用于 OSI/RM 和 TCP/IP 这两种体系结构的。

3.2 OSI/RM 体系结构

OSI/RM 是第一个标准化的计算机网络体系结构，它主要是针对广域网通信（也就是不同网络之间的通信）进行设计的。

3.2.1 OSI/RM 的七层结构

OSI/RM 将整个网络通信的功能划分为七个层次，由低到高分别是物理层（Physical Layer）、数据链路层（Data Link Layer）、网络层（Network Layer）、传输层（Transport Layer）、会话层（Session Layer）、表示层（Presentation Layer）、应用层（Application Layer）。在这七层结构中，源端的数据通信流程是由上而下，而目的端的数据通信流程是由下而上，如图 3-7 所示。

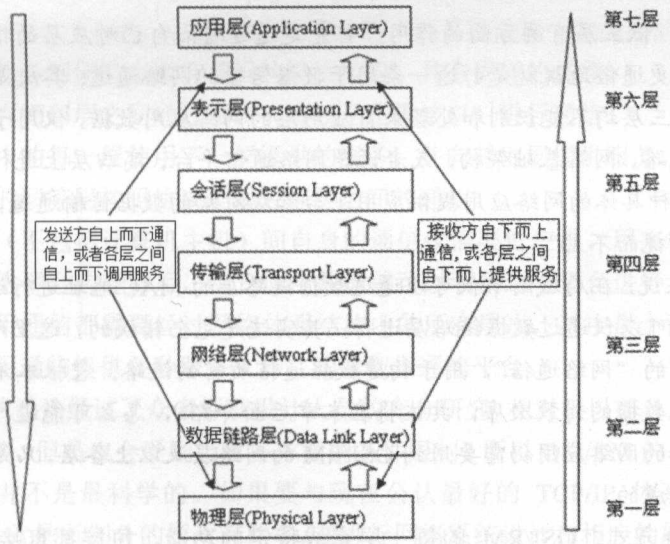


图 3-7 OSI/RM 七层参考模型

在 OSI/RM 的七层体系结构中，低四层（从物理层到传输层）定义了如何进行端到端的数据传输，也就是定义了如何通过网卡、物理电缆、交换机和路由器进行数据传输；而高三层（从会话层到应用层）定义了终端系统的应用程序和用户如何彼此通信，亦即定义了如何重建从发送方到目的方的应用程序数据流。

如果按 ARPAnet 网络中的体系结构划分，则是把 OSI/RM 的七层结构分成低三层和高四层，低三层负责创建网络通信所需的网络连接（面向网络），属于“通信子网”部分，高四层具体负责端到端的用户数据通信（面向用户），属于“资源子网”部分。OSI/RM 结构中各层的功能如图 3-8 所示。

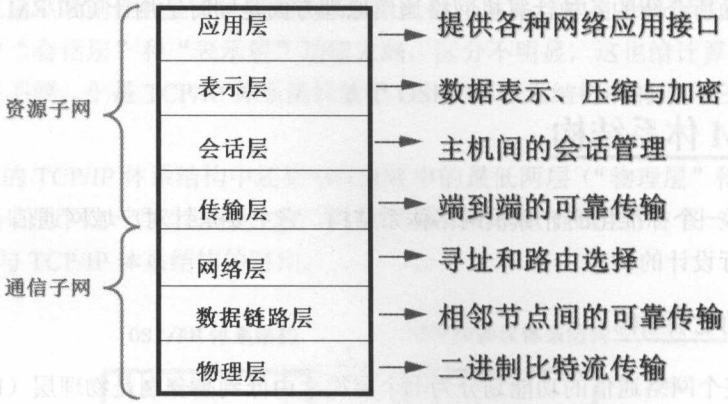


图 3-8　OSI/RM 各层基本功能



在 OSI/RM 中，低三层有两方面的作用：首先是通过它们自己对应层的信息交换构建数据通信所需的网络平台，更通俗地说就是打通一条用于数据传输的网络通道；其次是为来自上层的数据提供传输通道。但低三层均不能识别和处理来自应用层的网络应用数据，仅用于为用户的网络应用数据通信提供通信线路、网络基础架构，或者说是网络通信平台。高四层上进行的才是真正面向用户的网络应用，为各种具体的网络应用提供应用平台和端对端的数据传输通道，对低三层所构建的网络平台可以说是“视而不见”。

我们经常听别人说，在局域网中仅可以通过数据链路层的 MAC 地址进行通信，很多人就误解了，认为网络应用也可以仅通过数据链路层进行。其实这是完全错误的。这里所说的“通信”其实是仅指二层设备之间的“网络通信”，用于构建数据通信所需的链路。它根本不能识别网络应用的用户数据（这涉及到数据的封装次序，具体将在本章后面介绍），怎么可能进行网络应用通信呢？在局域网中进行具体的网络应用仍需要用到 OSI/RM 的网络层及以上各层，只是这些高层是由用户主机的操作系统来完成的。

无论是哪种划分方式，OSI/RM 的每一层完成特定的功能，每层都直接为它的上一层提供服务，同时又调用它的下一层所提供的服务。所有层次都互相支持，在发送端网络通信是自上而下进行的（也就是自上而下调用服务），在接收端网络通信是自下而上进行的（也就是自下而上提供服务），但双方必须在对等层次上进行通信（这就是对等通信原理，具体在本章后面介绍）。当然并不是每一通信都需要经过 OSI/RM 的全部七层，要视具体通信的类型而定，有的甚至只需要双方对应的某一层即可，如物理层中的物理接口之间的转接以及中继器与中继器之间的连接就只需在物理层中进行即可，而网络层中路由器与路由器之间的连接则只需经过网络层以下的三层即可。



无论哪一种计算机网络体系结构，也无论是体系结构中的哪一层，都不是针对具体的设备或者具体的软件而言的，而是针对每层中所要实现的网络服务功能来划分的。因为每一层所代表的是一组网络功能，而实现某一个功能又可以有许多不同的软/硬件方案。如物理层上就可以有许多不同的传输介质（如同轴电缆、双绞线、光纤等）和网络设备（如集线器、中继器），当然还有许多对应的通信协议。其他各层也一样。计算机网络中的软/硬件是计算机网络通信和数据传输的实体，也就是网络任务的具体执行者。当然各层的实体都不一样，具体将在后面对应章节中介绍。

有了这样一个结构模型，就把整个计算机网络软硬件技术和设备串起来了，所有软硬件技术都围绕在这个中心周围。

3.2.2 OSI/RM 的层次划分原则

OSI/RM 的七层划分以及每层的功能定义都不是随意的，都是经过标准设计人员对计算机网络通信原理、通信原理中各个流程之间的相互依赖关系，以及充分考虑了计算机网络软/硬件产品开发的方便性等诸多方面做了仔细、细致的研究得出的，都是要遵循一定原则的。具体原则如下：

- 同一层中的各网络节点都有相同的层次结构，具有同样的功能。
- 同一节点内相邻层之间通过接口（可以是逻辑接口）进行通信。
- 七层结构中的每一层使用下一层提供的服务，并向其上层提供服务。
- 不同节点的同等层按照协议实现对等层之间的通信。
- 网络设备（不包括计算机主机）间自身的通信仅需要低三层，用来构建数据通信的网络平台，网络平台构建好后，用户应用数据就可以利用这个平台进行各种网络应用，但所有网络应用通信都需要经过网络体系结构中的所有层次，其中最上面的四层用来为用户的网络应用通信提供各种服务支持，构建数据通信平台。

尽管 OSI/RM 的设计经过了众多标准设计人员的充分研究，相对于以前各厂商提出的体系结构来说有相当的优越性，但是也会受到当时各方面的条件限制，所以 OSI/RM 的七层结构划分从现在来看有些地方仍然并不是最科学的。如果要与现在公认最好的 TCP/IP 体系结构来比较的话，OSI/RM 的主要不足就是所划分的层次有些多，在进行网络系统设计和相应的软/硬件产品开发时仍然觉得比较麻烦。另外，像“会话层”和“表示层”单独划分的意义并不大，因为它们的用途并不像其他层次那样明显。也正因为如此，在后面的 TCP/IP 协议体系结构中不再有这样两层了。

3.3 TCP/IP 协议体系结构

TCP/IP 协议体系结构（也称“TCP/IP 协议参考模型”）是专门针对使用 TCP/IP 协议簇的广域计算机网络而开发的，可以说是 OSI/RM 的改进版本。但绝不能简单地认为是改进版，因为它与

OSI/RM 所针对的网络类型存在较大区别，所以这两种体系结构中各层所采用的通信协议以及功能实现原理上都存在非常大的差异，这一点在本书后面的章节中都会有相应的体现。现在我们常用的通信协议绝大多数不是很适用于 OSI/RM，而是适用于 TCP/IP 协议体系结构，因为它们都是应用于 TCP/IP 网络中。

3.3.1 TCP/IP 体系结构层次

TCP/IP 协议体系结构最初只划分了四层，从高到低分别是：应用层（Apllication Layer）、传输层（Transport Layer）、网际互连层（Internet Layer，也有称“互联网层”的）和网络访问层（Network Access Layer，也有称“网络接入层”“网络接口层”或“主机－网络层”的）。虽然只有四层，但它却包含了 OSI/RM 中的所有七层功能，同样包括了局域网和广域网通信所需要的全部功能。图 3-9 描绘了 TCP/IP 协议体系结构与 OSI/RM 层次间的关系。

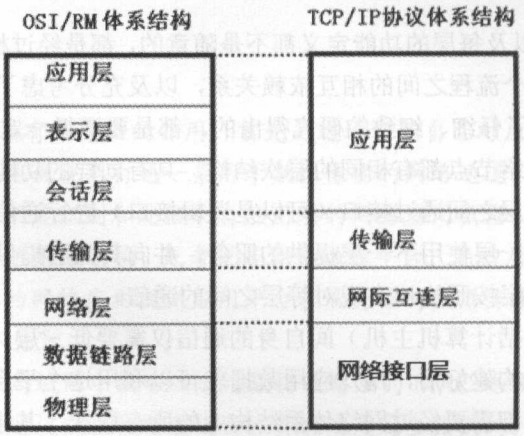


图 3-9　TCP/IP 协议体系结构及与 OSI/RM 体系结构的比较

从图中可以看出，在 TCP/IP 协议体系结构中对原来 OSI/RM 的七层结构进行了进一步简化，主要体现在以下两个方面：

（1）把原来的“物理层”和“数据链路层”合并为一层，即“网络接口层”，它提供局域网中通信连接的功能。

（2）合并了原来 OSI/RM 中最高的三层，成为新的“应用层”。因为事实上，在 OSI/RM 中会话层和表示层的功能都非常单一，完全可以合并到应用层中。

其他两层，“传输层”与 OSI/RM 中的功能划分是一样的，而“网际互连网”也与 OSI/RM 的“网络层”实际上是一样的，只不过名称不一样而已。但这仅是从功能划分上来说的，实际上这两个体系结构是存在相当大差异的，因为 OSI/RM 是开放型的标准，适用于所有类型网络设计参考，而 TCP/IP 协议体系结构是专门针对 TCP/IP 网络的。

但在实际应用中，许多计算机网络软/硬件产品开发商和用户都感觉到，把 OSI/RM 中的“物

理层”和“数据链路层”合并成“网络接口层”并不很好，所以现在通常采用如图 3-10 所示这样的五层网络体系结构。它综合了 OSI/RM 和 TCP/IP 协议两种体系结构的优点，同时克服了这两种体系结构的不足。本书也将以这种目前广泛建议的五层体系结构进行介绍。

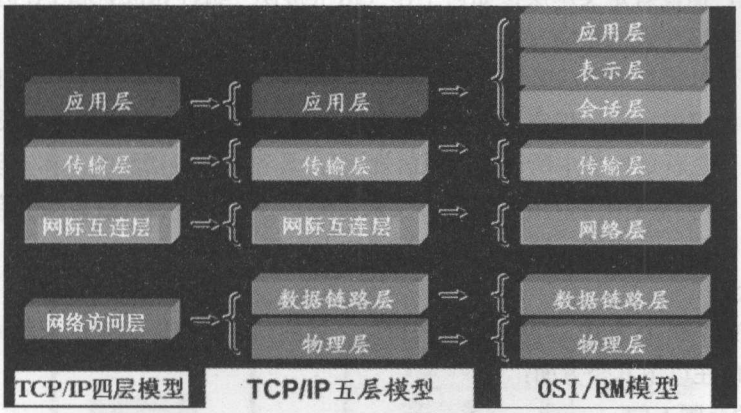


图 3-10 广泛建议的五层网络体系结构

3.3.2 OSI/RM 和 TCP/IP 体系结构的比较

TCP/IP 协议体系结构是在参考 OSI/RM 的基础上专门针对 TCP/IP 网络而开发的体系结构，所以它既继承了 OSI/RM 的基本模型结构和层次划分思想，又针对了特定的 TCP/IP 网络，更加具体化，更加具有可操作性。本节将具体介绍这两种体系结构的主要异同方面。

1. 相同之处

总体而言，OSI/RM 和 TCP/IP 协议体系结构主要具有以下几方面的相同或相似点：

（1）层次结构划分思想相同。

这两种体系结构都是以协议栈（不同协议形成的层次结构）为基础进行层次结构划分的，并且协议栈中的协议是彼此独立的。这样做的好处是可以大大简化各种网络协议程序的设计，只需要为不同协议程序提供关联的程序接口即可。

（2）总体层次结构相似。

在这两个体系结构中，虽然总的层数和对应层次的名称都有所不同，但总体层次结构还是极其相似的。TCP/IP 协议体系结构中的“网络接口层”对应了 OSI/RM 最低的“数据链路层”和“物理层”这两层，TCP/IP 协议体系结构中的“应用层”包括了 OSI/RM 最高的“会话层”“表示层”和“应用层”这三层，OSI/RM 中间的“网络层”虽然与 TCP/IP 协议体系结构中的“网际互连层”在名称上不一样，但功能却是完全一样的，至于“传输层”两种结构都是完全一样的。

在这两种结构中，“传输层”以下都属于“通信子网”部分，用来构建通信网络；而“传输层”及以上各层都提供了端到端的与网络无关的服务，属于“资源子网”部分。

（3）核心组件一样。

这两种体系结构中都定义了“服务”“接口”和“协议”这3个重要核心组件。

- “服务”也就是以本书后面各章说的“服务原语”定义了各层应该做些什么、要提供哪些功能，但没有定义本层该如何工作（由“协议”组件完成）和上一层该如何访问本层（由“接口”组件完成）。
- “接口”也就是本书后面各章中所说的SAP（服务访问点），为对应的上层提供了获取本层服务的逻辑接口，规定了有哪些参数可以使用和使用这些参数的结果是什么。
- “协议”也就是标准中所说的“通信规程”，是各层服务功能的具体实现者。当然，同一服务在不同网络中可以由不同的协议来实现。各层中的协议各自实现自己的功能，并不影响其他层，任何的一层，它只为相邻的上层提供服务。

2. 不同之处

在看到OSI/RM和TCP/IP协议体系结构相似之处的同时，也要看到这两种体系结构的许多不同之处，主要表现在以下几个方面：

（1）适用的范围不同。

OSI/RM是在标准化协议发明之前就已产生了，所以OSI/RM不偏重于任何特定的网络类型，具有最广泛的理论上的参考性，是一个理想化的模型。而TCP/IP协议体系结构则相反，它是在TCP/IP协议簇先出来了（而且仅适用于TCP/IP类型网络），然后开发这个模型的人员再针对这些协议进行功能分层和描述，所以与协议的关系非常紧密，最具有好的实践性。

（2）划分的层次结构不同。

这两种网络体系结构在层次划分上的不同在前面已有介绍，主要体现在两个方面：一是TCP/IP协议体系结构中没有“会话层”和“表示层”，因为事实已证明这两个层次并没有太独立的用途，所以最后取消了，合并在“应用层”之中；二是在OSI/RM中“物理层”和“数据链路层”的功能在TCP/IP协议体系结构中合并到了“网络接口层”中，尽管实际上在TCP/IP协议体系结构中对这个层中的具体功能并没有明确规定，但实际上这层的功能就是OSI/RM最低两层的功能。当然，事实上后来仍按照OSI/RM那样进行了拆分。

（3）支持的网络通信模式不同。

OSI/RM的网络层同时支持无连接和面向连接的网络通信（它不仅支持TCP/IP协议网络中无连接的IP网络协议，同时支持NetWare SPX/IPX网络中的面向连接的SPX服务等）；TCP/IP模型的网络层只提供无连接的服务（因为它只支持IP这种无连接的网络层协议）。

（4）包括的通信协议不同。

OSI/RM是一种开放型的，希望尽可能适用于所有类型计算机网络的理想化体系结构模型，所以它里面所包括的通信协议不仅非常多，而且类型非常复杂，适用于各类网络的都有。尽管TCP/IP网络也是在OSI/RM的设计范围内，但现在TCP/IP网络中的通信协议是专门针对具体的TCP/IP协议体系结构而开发的，所以更具有TCP/IP协议体系结构的特点。

3.3.3 局域网体系结构

目前计算机局域网标准是由 IEEE 发布的，所以其以太局域网体系结构最终也是由 IEEE 发布的，对应的标准为 IEEE 802.1a，如图 3-11 所示。

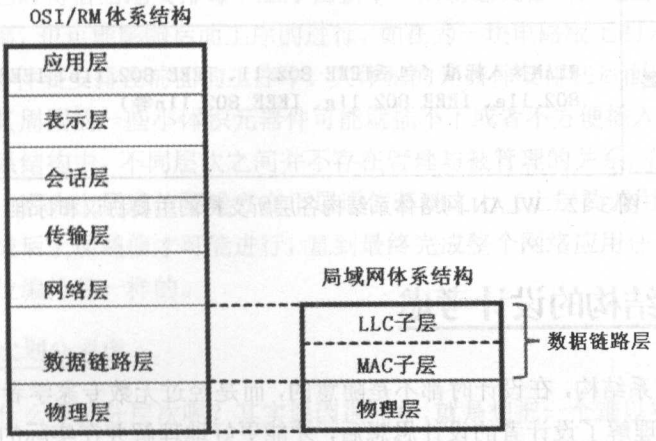


图 3-11 IEEE 802.1a 有线局域网体系结构

从以上局域网体系结构可以看出，总体来说是仅包括了 OSI/RM 或者五层 TCP/IP 体系结构中的最低两层（“物理层”和“数据链路层”），因为这仅是针对局域网的体系结构，而局域网中不涉及网络之间的通信，不需要进行网络层寻址，因此不需要体现上面各层。

【注意】这里所说的局域网只需要最低的“数据链路层”和“物理层”仅是指在局域网内部建立网络连接（交换机设备之间的连接）时只需要这两层，但在具体的网络应用中仍需要全部的各层，因为目前的计算机网络应用都是 TCP、IP 协议类型的，仍需要用到网络层的 IP 地址进行寻址，也需要用到传输层的 TCP 或 UDP 端口号来区分不同应用套接字。只不过，在局域网内部的网络应用，网络层以上的各层功能是由计算机主机的操作系统和网络应用软件完成的，在计算机主机中包括了体系结构中全部的各层功能。

另外，从图 3-11 可以看出，在局域网体系结构中对“数据链路层”进行了细分，分成了“介质访问控制”（Media Access Control，MAC）子层和“逻辑链路控制”（Logical Link Control，LLC）子层。这是因为以太网和 WLAN 属于广播类型的网络，存在介质争用现象（广域网中通常是属于点对点网络，通常不存在介质争用）。其中的 MAC 子层就主要是用来解决介质争用和局域网内部物理地址寻址的，而 LLC 子层担负着 OSI/RM 或 TCP/IP 体系结构中的“数据链路层”的功能。

目前分成了两大系列：一种是各种以太网标准构建的有线局域网，另一种是以各种 WLAN 标准构建的 WLAN 局域网，但它们均遵从图 3-11 所示的局域网体系结构，不同的只是各层所用到的协议、标准不一样。如图 3-12 所示是 WLAN 网络中局域网体系结构所用的一些主要 WLAN 协议或标准。



图 3-12　WLAN 网络体系结构各层所支持的主要协议和标准

3.4　网络体系结构的设计考虑

无论哪种网络体系结构，在设计时都不是随意的，而是经过无数专家学者充分认证和谨慎考虑得出的。我们只有在理解了设计者的设计思想后，才能更好地理解并在实际的网络系统设计、网络故障排除中应用这些设计思想。

3.4.1　体系结构的设计考虑

“网络体系结构”是一种概念上的蓝图，描述了整个网络的层次结构和基本的数据通信规则。实际上就是描述了整个网络中两节点间实现有效通信的所有过程，然后将这些过程划分为逻辑上的组，而这些组就是网络体系结构中所说的“层”。

网络体系结构设计时的考虑其实与一个公司在设计组织架构时要充分考虑整个公司的生产、经营流程、所需职能部门，以及各部门间协调效率等各方面的需求是一样的。

在为新公司设计组织架构时首先会考虑公司在生产、经营过程中需要进行哪些生产、经营活动，而这些活动又需要由哪方面的人才来完成，按什么顺序完成。而这里的“谁”不是指具体的哪个人，而是指一类人，也就是后面要设计的各个职能部门。当然，部门的设置也不是随意的，尽可能把相同功能范围的工作分配在同一个部门中，以提高部门内部的工作效率和部门间的协调效率。这里的“部门”就相当于我们所讲的网络体系结构中的“层”，体系结构中的层次划分也不是随意的，也是要对整个通信流程所要实现的各项主要功能进行宏观上的划分，而不是把每项功能单独划分一个层次。

另外，在公司中为了保证工作的顺利进行，每个部门内部的员工自然是紧密协调，但任何一个部门都不可能完全独立地工作，因为它们只是整个公司组织的一部分，必须要与其他部门协同工作，所以各部门都要有负责与上下级部门之间沟通的负责人，这就相当于网络体系结构中的层与层之间相互通信的逻辑接口，称为 SAP（访问服务点）。

既然计算机网络体系结构是分层的，那么不同层次之间必定有一个高低之分。就像一个公司中许多部门之间也是有权限高低之分的。但在体系结构中所讲的层次高低并不是从管理角度来理解的，可以说是仅从通信流程，或者说是服务调用关系上来划分的，主要是“谁先完成，谁后完成”的问题。就像在设计公司组织架构时要考虑命令上传、下达执行部门的先后次序一样。也像在工厂中为某个产品编排工艺时得恰当地安排每个工序在整个产品制造流程中的位置，否则一方面可能影响整个产品的制造效率，也可能影响后面工序的进行。如在为一块电路板上的元器件插入编排工艺时，要求小体积的元器件要安排在前面的工序中，大体积的元器件安排在后面的工序中，否则在插完大体积元器件时，其周围的一些小体积元器件可能就插不下或者不方便插入了。

在计算机网络体系结构中，不同层次之间并不存在管理与被管理的关系，但计算机网络通信还是有其必需的流程的，因为上层通信要建立在下层通信基础之上（上层要调用下层的服务），只有下层通信建立工作完成后上层通信才可能进行，直到最终完成整个网络应用任务。这就与工厂生产流水线的各个工序岗位编排是一样的。

3.4.2 体系结构的层次划分考虑

在体系结构中为什么要划分层次呢？其实原因很简单，就是想把一个难以实现的复杂问题分解成多个容易实现的小问题。这在我们学习时也一样，要真正把计算机网络专业学好不是一件容易的事，要学的知识和技能太多，涉及面太广。如果没有一个全局的观念，不采取分模块学习的方法，是很难学得系统、学得全面的。学习时，如果我们把要学的知识和技能分成一个个比较小的模块，然后采取“个个击破”的方式，就可以比较快地全面掌握计算机网络专业所需学习的知识和技能。但是我们的学习模块划分也要科学、合理，否则达不到提高学习效率的最终目的。

计算机网络体系结构是为所有网络通信提供结构标准的，必须要充分考虑各种复杂的网络通信流程，所以其中的层次划分肯定不是随意的，而是科学家们对各种复杂的计算机进行严格的认证，再经过深思熟虑后得出的。

那么这些网络结构中的层次是依据什么来划分的呢？主要还是针对不同网络环境中用户双方进行网络通信的流程或者通信原理来对不同网络功能进行分层的，就像一个产品的生产流水线划分多个生产、组装车间一样，每个车间完成一部分的工作，直到最后完成整个产品的生产和组装。

另外，体系结构中的层次划分还要考虑通信效率和可行性等诸多因素，既要使整个网络体系结构尽可能简单，又要确保各层的功能和不同层次之间的协商容易实现。如生产一个比较简单的产品（如组装一台小风扇），它既可以只设一个工序，由一人来完成所有的工作，又可以细化分成两个，甚至更多工序来协同完成。这就需要综合考虑生产效率和生产成本等多个方面的因素了。太粗了会影响生产效率（毕竟一个人做的速度有限），太细了不仅生产成本会大大增加（人工和设备费用比较高），也会影响生产效率（浪费在中间工序和产品生产流程上的时间比较多）。网络体系结构层次的划分也是一样的。

如在局域网体系结构的设计上，因为是专门针对局域网内部的网络通信进行设计的，就是对一个局域网内部（不涉及网络寻址）的网络通信构建基础的网络通信平台，而用户的具体网络应用所

需的网络层及以上各层功能完全可以直接通过局域网内部的用户主机操作系统来实现，所以也就只有“物理层”和“数据链路层”这两层了。

OSI/RM 和 TCP/IP 协议体系结构主要是针对不同网络之间的互联和通信而设计的，中间会有许多网络层或以上各层设备连接多个不同的网络，所以它们就不可能只有物理层和数据链路层，还得考虑网络层的寻址（需要网络层），在不同计算机网络中建立通信连接和数据传输通道。

至于为什么在有了可包括所有层次功能的主机系统后还要专门设立传输层和应用层，主要是出于一些特殊的网络应用需求，因为在网络设备（如交换机、路由器、应用层网关等）支持了这两层后，就可以基于这两层更好地实现网络应用控制（如基于传输层端口、应用层报文内容的过滤）。而且即使传输层和应用层在主机系统中实现也是体系结构的体现，主机中的操作系统也是基于体系结构而开发的。但事实上，在绝大多数网络中，交换机、路由器这些设备是不支持传输层和应用层功能的，这两层的功能基本上都是依靠主机系统来完成的。而且从计算机网络通信原理来看，传输层和应用层的报文都是在端到端构建的传输通道中进行传输的，不涉及中间设备，中间设备只需要为传输通道的建立搭建好网络通信平台即可。

3.4.3 主机系统层

前面我们多次提到了“主机系统层”，在网上也经常提到这个概念，那么它到底是个什么样的层次，它与计算机网络体系到底有什么关系呢？

前面我们说到，在局域网内部的通信是无需三层设备的，在局域网体系结构中也只有“物理层”和“数据链路层”。此时，可能马上有读者会问，难道在局域网内部就没有网络应用不需要进行数据传输，不能使用 IP 地址这类三层功能吗？答案大家都知道是否定的，因为进行网络应用和数据传输是任何计算机网络（当然也包括局域网）的最基本用途，否则计算机网络就没有存在价值了。其实这里我们要充分认识局域网中各计算机的角色了。

现在计算机中安装的操作系统都是网络操作系统，都具有网络功能，可以提供 OSI/RM 体系结构中的各层功能。其中物理层和数据链路层是由物理设备——网卡、传输介质来完成的，其他上面各层均是由操作系统软件的各对应功能模块来实现的。这样一来，在同一个 IP 网段的主机间在经过中间的网络设备时就没有必要经过三层的路由寻址，而所用的 IP 通信完全可以由各主机操作系统中的网络层功能模块来实现，因为现在的计算机操作系统支持 IP 地址配置和 IP 协议的网络应用。至于在传输层所建立的传输通道也完全可以由源主机和目的主机系统之间的传输层功能模块来实现，在操作系统中也是安装了传输层的 TCP 和 UDP 协议的。事实上，在两个远程网络间的通信中，中间设备（如路由器）一般也都只支持 OSI/RM 体系结构的下三层，高四层功能基本都是由各计算机提供的，如图 3-13 所示。

至于应用层，在主机操作系统中就更能全面支持了，毕竟网络应用主要是基于软件的，如 DHCP、DNS、HTTP、E-Mail、FTP 等服务器功能就全面包括了。尽管在局域网中仍然要进行基于 IP 地址类的网络通信和网络应用，在局域网中也要发布各种应用服务器等，但在局域网设备（不包括主机）间的通信中是不需要借助三层（网络层）和以上层次进行通信的，直接利用数据链路层

中的 MAC 地址进行寻址（在二层交换机上会建立 MAC 地址表）来建立数据链路层的连接。对于来自主机应用层的数据也必须经过完整的体系结构传输、协议封装和解封装流程。具体的数据传输、协议封装和解封装原理将在 3.6.1 节中介绍。

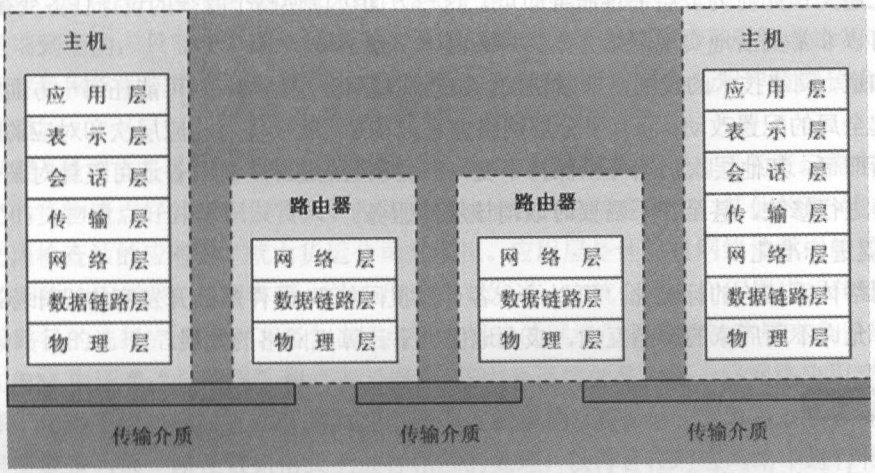


图 3-13 远程网络互联的基本网络结构

3.4.4 体系结构分层的好处

我们已经知道，计算机网络体系结构是分层的（但是需要注意，网络体系结构只是一个逻辑上划分的结构，真正起作用的还是各个网络实体，也就是网络设备软/硬件系统和用户进程等），但这样分层的体系结构到底有什么好处呢？综合起来有以下几个主要方面：

（1）便于方案设计和维护。

这是分层网络体系结构最重要的优点。对于一个复杂的网络通信过程，如远程的计算机或网络访问，在通信过程中可能要经过许多中间节点，也可能要进行多种信息格式的转换，需要多种协议来共同完成。如果没有一个分层结构，进行网络系统设计时就没有了思路，到底该设计哪些功能、每个功能的实现又需要依赖哪些协议和服务、如何实现这些服务功能等都不是很明确。

有了分层结构后，在进行网络系统设计时，程序开发者就可以根据体系结构来理解计算机网络通信的流程，并知道在每一层中需要实现哪些类型的功能。这时，负责各层功能的程序开发者可以选择自己最擅长的语言或技术来进行开发，而不用管其他层的功能是如何开发、实现的。然后再通过一些服务接口即可实现一个完整的网络通信任务。这样一来，就可以把一个大的网络通信任务分解成少数几个模块，然后采取个个击破的方法，很容易就可以实现复杂的网络通信方案。

在维护方面也一样，如果没有分层结构，一旦出现网络故障，就可能要对整个网络的各方面进行故障分析和排除，就像大海捞针一样，毫无目的可言，显然工作量很大；而有了分层结构后，我们就可以根据不同层次的功能特点和实现原理，以及故障特点来专门针对某一个或少数几个层次进行对应的故障分析和排除，显然容易多了。

（2）各层相互独立，技术升级和扩展灵活性好。

在网络体系的分层结构中各层是相互独立的，也就是没有管理和被管理的区分，也互不影响，每一层都不需要知道它的上下层是如何工作和进行功能实现的，仅需要上层知道该如何调用下层的服务，而上层又该如何为上层提供服务即可。这一方面可以使整个复杂的设计任务变得比较简单，另一方面可以非常灵活地实现网络方案扩展或技术升级。

我们知道，网络技术的发展是非常快的，如果不采用分层结构，则可能任何一方面的技术更新都可能引起全局的配置改动；而有了分层网络体系结构后，就仅需对相应层次和对应的服务接口进行技术更新即可，其他层次仍然可以保持不变，不受该层的影响。同样，还可仅针对某一个或几个层次的服务进行修改，甚至在不需要时取消该层的服务。

（3）促进标准化。

通过网络体系结构的标准化，可以统一各计算机网络软/硬件产品开发商的设计标准，实现协同开发，并允许不同厂家的产品互通，极大地促进了计算机网络的发展。



在进行网络体系结构分层时应使每一层的功能非常明确，而且比较容易实现。如果层次划分得太少，则会使每一层的协议太复杂，不利于实现；相反，如果层次划分得太多，每一层的功能太单一，层与层之间的接口太多，同样会影响网络通信效率。这与我们在设计产品生产工艺时的考虑是一样的。

3.5 例说体系结构各层主要用途

本节将以最常见的生活事例来诠释 OSI/RM 计算机网络体系结构中各层的主要用途（其他体系结构对应层次的功能类似，参照即可），以便能更加容易理解。当然或许里面有些比喻并不是很贴切，但只要对理解这些层次的功能有所帮助就达到了目的。

3.5.1 各层角色的类比

如果把计算机网络体系结构的各个层次与我们现实生活中交通运输行业的各部门进行类比，则可以像下面所述来理解。

“物理层”就相当于各个基建公司，负责基础设施的建设，如修路、搭桥。因为物理层的主要用途就是为各种计算机网络通信提供最基础的通信线路，负责把“路”修通。

“数据链路层”就相于各省市的交通管理局，负责本地区交通法规的制订和交通管理。因为数据链路层是基于网络分段的，不同网络所运行的数据链路层协议可能是不同的，就像不同地区负责道路基建、维护和管理的单位、法规要求都不一样，不能越界管理。

“网络层”就相当于国家的交通部，负责全国主干道修建计划和实施，制订适用于全国的交通

法规，最终的目的就是构建一个可通达全国的交通道路网。这部分的工作地方交通管理部门是无法实现的，因为地方法规只能作为全国交通法规的补充，不能替代和逾越。网络层的最终目的也是把各个原来分离的网络连接起来，最终实现各个网络的互联互通。

“传输层”就相当于各个物资运输公司，负责把用户的货物及时送达目的接收者。物资运输公司的货运是端到端的，只需要考虑从源端到目的端的货运路线选择，而不用管途经各道路的建设。传输层在计算机网络通信中也是起到一个端到端数据传输的作用，只是为用户数据传输构建一个虚拟的传输通道。

“应用层”就相当于各个需要进行货物运输的用户。用户只需要把自己的货运需求告诉货运公司，对自己的货物清点好并标识发货人、收货人的地址和人名信息即可，其他的事（如运输路线的选择、与交通检查站的应对等）就由货运公司来承担。应用层是计算机网络体系中的最高层，是一个个具体的网络应用需求，只要有用户的应用数据要发送，它就会把它向下层进行传输，至于最终数据的传输路径的选择完全是由下面负责网络平台搭建的网络层和数据链路层来负责的。

在 OSI/RM 中还有“会话层”和“表示层”，其中会话层就是为每一路网络应用建立一个唯一的会话标识。相当于用户同时发送许多到达不同地方的货物，要做好记录和标识，以免搞混、搞错一样。表示层就是为每一路在数据格式上进行相应的处理，以符合在对应网络中进行传输。就像有时用户发送的货物比较特殊，如是易碎品，这时就需要对货物进行特殊的包装。但因为这两层的功能都比较单一，所以在 TCP/IP 体系结构中合并到了应用层，因为这本来就可以都由用户自己进行处理。

3.5.2 各层主要作用的类比

在理解了计算机网络体系结构各层的主要角色后，下面进一步通过类比的方法来理解各层在计算机网络通信中的主要作用。

1. *物理层*

物理层是 OSI/RM 和 TCP/IP 体系结构的最低层，负责把网络中的各个设备通过相应的通信协议连接起来，为网络中的所有数据转发提供最基本、最必需的通信路径。简单地说，物理层就是用来构建计算机网络通信和数据传输路径的，所有数据均必须在物理层构建的传输路径中进行传输。就像我们现实生活中的行走、旅行，必须得有路才行，这些路就包括普通公路、铁路，或者航空、航海的航线。

计算机网络体系结构中的层次划分都是从整个计算机网络通信这个宏观角度进行考虑的，并不会针对具体的设备来定义。“物理层”的定义也不是针对具体的传输介质、设备和通信协议的，在不同类型的网络中可以有许多种选择，如物理层所用到的传输介质就可以有同轴电缆、双绞线和光纤等，而物理层所规定的网络接口的结构和形状也可以各种各样，只要能在某种网络中实现其特定的物理层功能即可。

不同的传输介质和设备选择必须要有对应的通信协议支持，而且这也决定了不同的选择有不同的物理层性能。就像路有好多种，如黄泥巴路、沙子路、水泥路、柏油马路、普通铁路、高速铁路

等一样，这些不同的路可以承载的重量和通过速率都不一样。不同的路连接起来就形成了我们到达某个目的地的完整路线。同样，计算机网络中的不同物理层相连也构成了双方的整条通信路径，但是这些不同设备所连接的传输介质不是直接相连的，而是数据从一条传输介质进来，经过相应处理后，再从本地设备的另一个端口所连接的传输介质中发出去。

如图 3-14 所示，PC1 要与 PC2 通信，则它们之间的通信路径就包括了由 PC1 到 LSW2 交换机的 Ethernet 0/0/2 接口之间的电缆、由 LSW2 的 Ethernet 0/0/2 接口到 LSW1 的 GE0/0/1 接口之间的电缆、由 LSW1 的 GE0/0/2 接口到 LSW3 的 Ethernet 0/0/1 接口之间的电缆和由 LSW2 交换机的 Ethernet 0/0/2 接口到 PC2 之间的这 4 段线路组成。

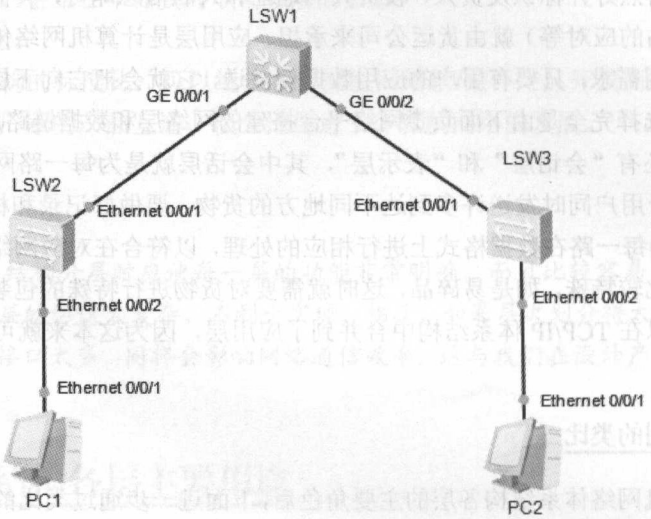


图 3-14　通信路径组成示例

另外，我们知道在各种道路中都会划分许多车道，在计算机网络体系结构中也有类似的“车道”，那就是我们通常所说的“信道”。信道的全称就是“信号传输通道”，是在传输介质上成功运行了对应的物理层协议后所建立的一个用于数据传输的逻辑通道。默认情况下，一条物理线路就是一条信道，但也可以通过各种信道复用技术在一条物理线路中划分出多条子信道，而每一条子信道可专门用于不同的应用通信，这样就提高了传输介质或传输带宽的利用率。有关信道复用的方式将在第 5 章中介绍。

2. 数据链路层

前面的“物理层”是为了构建数据传输通道所提供的基础线路，主要任务就是把各设备通过传输介质在电气上连接起来。设备在电气上连接起来后仍不能进行真正的数据传输，因为不同类型的网络所运行的通信协议不一样，设备间真正建立邻居关系还需要在两端运行相同的通信协议，这就是数据链路层要完成的工作了。

“数据链路层”的主要作用是在物理层构建的物理链路或传输信道（传输信号的通道）的基础

上通过运行对应的链路层协议使在链路两端设备间构建的数据传输通道适合对应类型协议封装的数据在通道中以“帧”为单位进行传输，为对应类型协议的数据传输构建一条逻辑链路。

“数据链路层”可理解为我们的市内公路+交通法规。之所以只能理解为市内公路，那是因为在网络中的“数据链路层”间的通信数据仅可以在同一 IP 网段内的各设备间进行转发；之所以还要加上“交通法规”，那是因为“数据链路层”所提供的不再是物理线路，而是在物理线路的基础之上通过专门的数据链路层协议（相当于市内交通法规）构建的逻辑数据传输通道。

3. 网络层或网际互连层

OSI/RM 中的“网络层”（或 TCP/IP 协议体系结构中的“网际互连层”）是为不同 IP 网段之间的数据转发提供路径选择，通过 IP 地址（也可以是其他网络层地址，要视具体网络类型而定）把数据包转发到目的节点，可以理解为交通网络中的“车站”“机场”“码头”。这涉及到一个选择下一站路径的问题，也就相当于我们要到某外地去旅行，到了车站、机场、码头后要选择乘坐哪趟车、哪趟飞机或轮船才能最快捷，成本最低。

网络层的这种寻址功能就是我们通常所说的“路由寻址”，就是选择哪条路径来到达下一个路由结点。通过不同的路径进行路由，在性能、成本上都可能不一样，就像我们选择不同的交通工具或者不同线路旅行时的效率和成本都不同一样。网络层的功能就相当于连接不同网络的桥梁，仅起到在不同网络间转发数据包的作用，最终数据还是要在目的网络的数据链路层进行传输，在到达下一个网络结点设备（如路由器）时再进行路由、转发。

另外，就像我们到达另外一个城市又得遵照另外城市的交通法规一样，当我们发送的数据包到达另一个网络时，同样要遵照这个网络中的链路规则，也需要有相应的链路层协议来对数据重新进行封装，以实现数据可以在对应链路层协议类型的网络中传输。

4. 传输层

“传输层”是在下面三层构建的网络平台的基础上专门为通信双方构建端对端（不是点对点）的数据传输通道，使通信双方就像直接进行数据传输一样。这个端对端传输通道是可以跨网络的，这与数据链路层所构建的仅用于局域网内部的传输通道是不同的。就像我们远隔重洋与自己的家人或朋友通电话时感觉就在国内，甚至就在自己身边一样。

传输层类似于国际航线，一条国际航线可能要经过几个国家，但国际航班飞机飞行时根本不用考虑经过了哪些国家的航线，因为这些事先在确立国际航线时就已经处理好，就像一条传输通道要经过几个网络不用管，因为网络层事先已准备好了通信路径。

5. 会话层和表示层

这两层仅在 OSI/RM 中单独划分，而在 TCP/IP 协议体系结构中是没有这两层的，那是因为表示层和会话层的作用太单一，合并到了应用层中。

“会话层”是为具体的用户应用建立会话进程（每个应用都有一个会话进程），是一个用户网络应用的协商过程，相当于车站、机场或码头中总调度人员所从事的调度工作，为每趟班车、航班和轮船分配一个独立的作业单号，以便管理。

“表示层”是对用户网络应用数据的具体解释，包括在网络通信时可采用的信息格式、可采用

的加密方式。这就相当于车站、机场、码头中为每一趟班车、航班、轮船所编写的具体文件，包括所采用的车型、机型、船型，以及所负责的运输公司、交接人员名单等内容，使接收该班车、航班和轮船的运输公司了解具体的详情。

6. 应用层

“应用层”是用户进行具体网络应用的层次，由各种网络应用软/硬件负责完成，是具体网络应用的发起者和接收者，负责接受用户的各种网络应用请求，并为他们分配对应的网络应用进程和所需资源。

应用层的作用就像车站、机场和码头的总调度人员一样，一旦接到运输请求先审查目前是否有足够的资源支持，确定后负责为每一班运输业务安排发送时间，配备相应的人员和设备及其他一切资源。

3.6 计算机网络通信原理

无论是以上哪种计算机网络体系结构，在通信原理上它们都有两个相同的方面：一个是在网络连接和数据传输流程方面，发送端是自上而下（也就是从高层到低层），接收端是自下而上（也就是从低层到高层）进行的，也就是数据通信原理；另一个是通信会话方面，双方都必须在逻辑上的对等层次上进行，也就是对等会话原理。如发送端的网络层只能与接收端的网络层通信，而不能直接与数据链路层或传输层进行交错通信。其他的层次也是一样的。

3.6.1 计算机网络通信流程

在各种计算机网络体系结构的网络连接和建立数据传输的流程中，发送端是把通信连接建立指令和用户应用数据从上层向下层传输的，直到最低的物理层；而接收端是把通信连接建立指令和用户应用数据从下层（从最低的物理层开始）向上层传输，直到与发送端发起通信的对等层。如图 3-15 所示是 OSI/RM 情形下数据的传输流程，其他体系结构的数据传输流程类似，只是在数据传输中流经的层次多少上有区别。

其实这也很好理解，毕竟网络通信连接和数据传输都不能凭空建立或进行。它必须通过计算机网络中最基础的也是网络体系结构中的最低层——物理层，通过传输介质来传递各种通信连接建立信号和数据。但在进行具体的数据传输前，必须先建立好相应的连接，可以是永久连接（如局域网中的连接和广域网中的各种专线连接），也可以是非永久连接（如各种拨号连接和串行线路）。就像我们上互联网都要拨号连接（专线连接的除外），打电话我们首先要拨号一样。再如，当我们要运输货物到外地，首先得找好运输货物的路径，联系好接收货物的人，在对方确认可以收货后才可以发货，否则路都不知道怎么走，对方都不想要这批货物，自然就不能发货了。

计算机网络体系结构中的这种源端自上而下，目的端自下而上的通信连接建立和数据传输流程与我们在公司中安排、完成一个具体任务的流程是一样的。

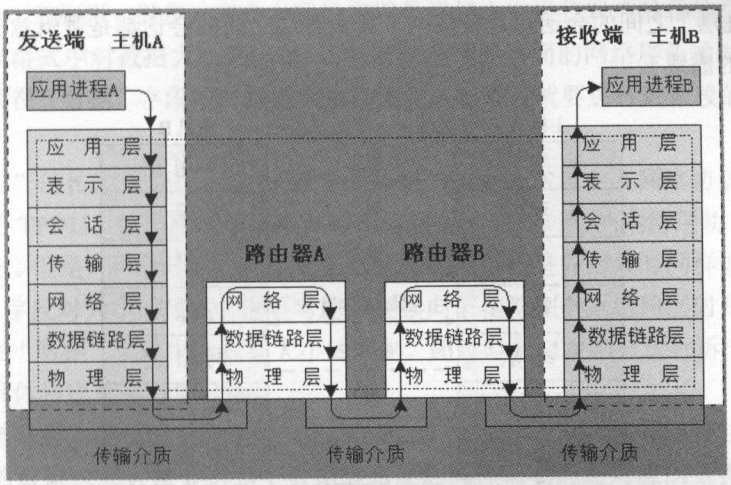


图 3-15 OSI/RM 数据通信原理示意图

现假设 A 公司的总经理要与 B 公司的总经理签个协议，则一般是按照这个流程进行的：①A 公司的总经理把这个要求向他的某下级部门经理交待；②该部门经理又会把这项具体的任务交待他下面的某个负责这方面工作的员工做好相应准备；③具体负责员工负责并与对方公司取得联系。

B 公司的执行流程与 A 公司的正好相反，因为这个任务请求最先是由 B 公司下面负责具体联系工作的员工收到的，他需要一级级地向上反馈。在他接收到 A 公司的这个请求后，会向他的直接上级部门经理反映，然后这个部门经理又把这个请求向他的总经理反映。

B 公司的总经理还可能要根据他具体的工作任务安排具体的签协议日程，然后把这些信息依次向下反馈到他公司中具体负责这方面的工作人员，再与 A 公司的具体工作人员沟通，A 公司人员接到 B 公司的这些信息后，又要依次向上传达到他们的总经理。在这其中可能要经过多次反复，最终完成整个 A 公司总经理与 B 公司总经理签协议事件信息的下达（在 A 公司）、上传（在 B 公司）任务。接下来的事就是两公司总经理之间他们自己的事了，这就是在本节后面要介绍的通信会话原理。

3.6.2 对等会话原理

通信双方的网络连接建立好后，就可以进行各种具体的网络应用和网络通信了，但这时的通信会话是建立在双方对等层次上的，也就是我们通常所说的“对等层”（Peer Layers）通信原理。一方面是因为只有双方是对等层次的会话才可能使用相同类型的协议，彼此才能“听得懂”，才有“共同语言”；另一方面，数据只有到达相同层数才具有相同的最外层协议封装，才可能识别本层次的相关地址信息，然后再根据这些地址信息来寻址；再一方面，在网络体系结构中，每一层是独立完成其工作的，其他层都是不干预、不了解的。如一方的物理层只能与对方的物理层直接通信，不可能直接跳到与对方的“数据链路层”或其他层进行对话。

同理，一方的网络层只能与对方的网络层通信，一方的传输层只能与对方的传输层通信，依此

类推。只有“物理层”之间的会话才是直接的，其他各层之间的会话都是逻辑意义上的，如图 3-16 所示（注意其中的虚线）。

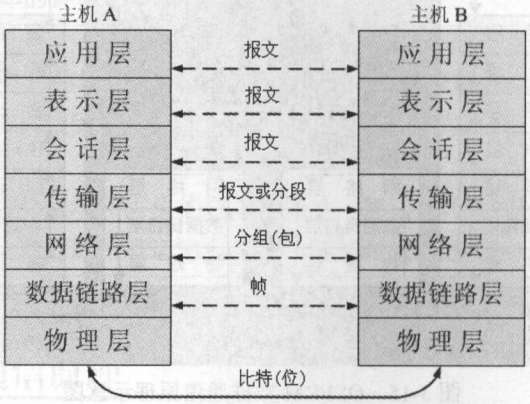


图 3-16　OSI/RM 对等通信原理示意图

计算机网络通信的这种对等会话原理可以理解为要级别相当的双方才能对话一样，主要考虑到这样才能有“共同语言”。如国与国之间的外交，总统/主席级的只能与总统/主席级的直接对话，总理级的只能与总理级的直接对话，部长级的只能与部长级的直接对话，不能越级，否则大家谈不到一起。

【注意】尽管必须遵守对等通信原理，但最终的数据通信流程还是要经过发起会话通信的层次下面所有的其他层次的，只不过，在具体的通信过程中，对于用户来说，下面各层是透明的，在网络连接建立好后，我们并不需要了解下面各层的工作情况。所以我们在进行具体的网络应用（如文件传输）时，似乎是直接把文件传到了对方。

3.6.3　各层的数据传输单元

在计算机网络体系结构中，每一层都运行着不同类型的通信协议，每一层在进行数据传输前都要通过这些协议对所传输的数据进行特定格式的封装（除“物理层”外），而且每一层封装的格式和数据单元 PDU（Protocol Data Unit，协议数据单元）都不一样。本节就以源端即数据发送端为例介绍在数据发送时经过每一层时所采用的数据单元，至于每一层数据的协议封装方式将在下节介绍。

（1）在会话层、表示层和应用层中是以实际的数据报文进行传输的。

在会话层、表示层和应用层中的数据都是以原始的数据单元格式进行传输，当然每经过一个层次都要由对应的协议进行封装。通常称之为应用数据报文。

（2）传输层比较特殊，在 OSI/RM 体系结构中传输层数据单元称为 TPDU（传输协议数据单元），而在 TPC/IP 协议体系结构中，TCP 是以数据段（Segment）为单位进行传输的，UDP 是以数据报（Datagram）为单位进行传输的。

来自上层的应用数据到达传输层时也要经过对应的传输层协议进行再次封装，形成一个个

TPDU 或者数据段、数据报。但是，来自上层的应用数据报文在进行对应的传输层协议封装时，不仅要考虑自身协议格式中对数据大小的限制，还要考虑继续向下面的网络层传输时是否会超过对应的网络层协议格式在“数据”字段中对数据大小的限制，超过时就要进行数据段的拆分，具体将在第 8 章中介绍。

（3）“网络层”的传输单位是“包”（或者“分组”），称为 NPDU（网络协议数据单元）。

来自传输层的 TPDU、数据段或数据报到达网络层也要经过对应的网络层协议进行再次封装，形成一个个数据包。但是，网络层的协议在对来自传输层的数据进行封装时同样既要考虑到自身协议格式中“数据”字段对大小的限制，也要考虑到继续向下传输到数据链路层时是否会超过对应的数据链路层格式在“数据”字段中对数据大小的限制，超过时要对数据包进行拆分，分解成一个个小的数据分组，具体将在第 8 章中介绍。

（4）“数据链路层”的传输单元是“帧”（Frame），称为 DPDU（数据链路协议数据单元）。

来自网络层的数据包或数据分组到了数据链路层后同样需要经过对应的数据链路层协议进行再次封装，形成一个个数据帧。因为在网络层向数据链路层传输数据前就已考虑了下面的对应数据链路层协议对数据大小的限制，而且数据链路层封装的数据帧最终是要在物理层以“无格式”、最原始的一位位比特流方式进行传输，所以这时数据链路层的帧封装就无需考虑其下面物理层协议对数据大小的限制了。

（5）“物理层”是以最原始的“比特”（bit）流格式传输的，或者说物理层的 PDU 就是“比特”，一位位地在传输介质的信道中传输。

以上各层的数据传输单元参见图 3-16。如果是其他网络体系结构（如 TCP/IP 协议体系结构、局域网体系结构），各层上传输的数据格式也是一样的，不同的只是少了一些层次而已。

3.6.4 协议封装和解封装

从上面的分析我们已经知道，在整个数据传输过程中，数据在发送端是自上而下逐层传输的，每经过一层（除“物理层”外）都要进行一次“协议封装”，对来自上层数据加上本层所使用的通信协议的头部（即“协议头”），其目的就是用以标识对数据在使用某通信协议时所配置的参数信息，特别是各层的“地址”信息。

以上是从发送端的数据传输流程进行分析的，数据在接收端是自下而上逐层传输的，数据每经过一层（除“物理层”外）都要进行一次“协议解封装”，是发送端“协议封装”的逆过程。协议解封装的目的是去掉来自下层数据原来所携带的下层协议头部，使数据原来在发送端与本层相同层次封装的头部信息能被识别，因为每种通信协议只能识别数据的相同协议头部信息，而且这些头部信息必须是在数据的最外层封装中，否则这些头部信息就会被当作“数据”字段进行处理。

【说明】数据的发送从哪层发起不固定，每一层都可以发送数据（不一定是来自应用层的数据），但是数据源自发送端的哪一层，在接收端最终也只会在相同层次进行处理（不会再向上传），这就是 3.6.2 节所介绍的对等通信原理。

如图 3-17 左侧箭头所示的顺序就是 OSI/RM 各层的数据封装流程，每向下传输一层都会在原

来从上层接收的数据的最前面再加上本层运行的协议的协议头（数据链路层中还要同时加上“协议尾”）。其中的 AH 为应用层协议头，PH 为表示层协议头，SH 为会话层协议头，TH 为传输层协议头，NH 为网络层协议头，DH 为数据链路层协议头（物理层为最低层，传输的是最小单位的 bit（比特），所以不需要再进行封装，所以没有“物理层协议头”），DT 为数据链路层协议尾（这是数据链路层所特有的，用于接收方对帧进行正确性校验）。

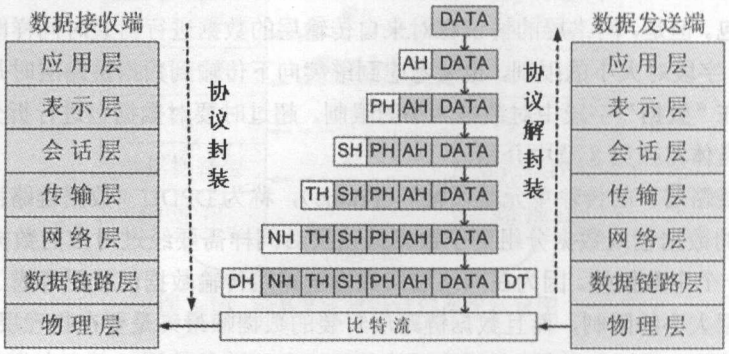


图 3-17　OSI/RM 各层数据封装流程

如图 3-17 右侧箭头所示的顺序就是“协议解封装”的流程，每向上传输一层都对数据中本层所封装的协议头进行剥离（数据链路层中还要同时剥离“协议尾”）。因为上层并不需要了解它的下层服务，所以当报文送到某一层时就会把用来标识它的下一层的协议头和协议尾去掉，还原该报文在发送端对应层时的协议头。其实加上协议头的作用可以理解为在发送端要一层层地加上一个指明到达目的网络或者下一链路地址信息的信封，而在接收端则要一层层地拆开信封，以获取数据继续向下游网络传输的地址信息。

TCP/IP 体系结构中的数据封装和通信原理也与 OSI/RM 类似，只不过没有 OSI/RM 这么多层次。如图 3-18 所示是两个中间隔了两个网段的主机（Ha 和 Hb）在 TCP/IP 网络中的基本通信流程，注意箭头的方向（假设通信是由 Ha 主机发起的）。

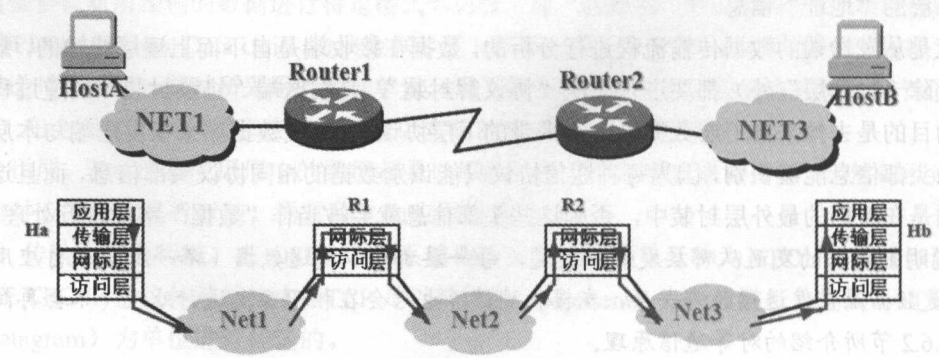


图 3-18　TCP/IP 网络中的数据通信流程示例

总体来说，计算机网络体系结构的基本通信原理是数据在发送端自上而下，而在接收端是自下而上传输的，整个通信过程必须依次经过通信发起层及以下各层，而不能跨越。但是各层只能与对方的对应层进行通信，而不能出现错位。

3.7 网络体系结构中的通信协议

说到计算机网络体系结构，就不得不说计算机网络中的通信协议（也就是在标准中常说的“通信规程”），因为这些通信协议就是计算机网络体系结构中不同层次中的具体功能实现者，也就是我们通常所说的各层所提供的“服务”。

3.7.1 理解计算机网络通信协议

通信协议对于计算机网络的重要性是不言而喻的，可以说没有通信协议就没有计算机网络，是计算机网络组成必需的一部分。当然，事实上任何一种计算机网络设备都不可能仅是一台硬件（俗称“裸机”），必须安装了对应的软件系统，而这些软件系统中只有包含了对应的通信协议才可能进行对应的网络通信或网络应用。

由于现在计算机网络种类很多，每一种计算机网络要进行最终的网络通信和网络应用就必须有一系列的各层通信协议来支撑。如我们现在常用的 TCP/IP 网络，在应用层常见的通信协议有 DNS、DHCP、HTTP、POP3、SMTP 等，在传输层有 TCP 和 UDP 协议，在网络层有 IP、ARP、ICMP、IGMP 等，在数据链路层有 IEEE 802.3、PPP、PPPoE、PPTP 等，在物理层有 RS-232、V.20、V.21、V.35 这样的接口标准。

那么什么是计算机网络通信协议呢？很简单，只要联系到我们平常与人家签协议是什么概念就可以理解计算机网络中的通信协议了。

在平常的工作和生活中我们与别人签订协议的目的是什么，就是约束双方或者多方的行为，达成对某一方面或者多个方面的共识。计算机网络通信协议也是这样的，它就是约束通信双方在利用某协议进行通信时必须遵从对应的规则和约定，以达成通信共识，否则彼此各不相认，还如何通信呢？就像一个只会中文，一个只会英文的两个人对话，肯定不能对话成功，因为他们相互都不明白对方在说什么。再比如，如果有几个人共同开发一个方案，结果他们各自使用的程序开发语言完全不同，而且不能相互兼容，这样最终肯定是不能形成完整的方案。所以，计算机网络通信协议可以理解为，为了使网络中的不同设备能进行协同的数据通信而预先制定的一整套通信双方相互了解并共同遵守的格式、流程和约定。

计算机网络的发展经历了漫长的探索过程，因此实际上并不是先制定好了统一的标准再研究网络的。在网络发展中有很多国际组织和跨国公司都在致力于网络通信协议的制定，并产生了多种计算机网络体系结构模型和网络通信协议，更多的组织和公司还在不断地参与开发和完善网络协议的工作。但需要注意的是，通信协议并不一定是标准，也可能仅是某个公司为他们的网络设备专门开发的通信协议，不能用于其他类型的计算机网络中，也不能与其他厂家的设备进行互联。

如果你对网络设备有所了解的话就一定清楚，Cisco（思科）有许多自己的协议，如 CDP（思科发现协议）、DTP（动态中继协议）、VTP（VLAN 中继协议）、IGRP（内部网关协议）和 HSRP（主机备份路由器协议）等，这些协议是不能在其他厂家的设备上使用的（除非得到思科的授权），也不能与其他厂家的设备互联。只有通过国际上的一些标准化组织（如 ISO、IEEE、IEC 等）认可后才能以标准形式发布，如 ISO/RM 本身也是一个协议，但因为是由 ISO 制定并发布的，所以成为了国际标准，还有 IEEE 的一系列以太网、城域网、WLAN 等协议也是公认的国际标准。

3.7.2 网络通信协议的三要素

网络体系结构中的各项功能服务都是通过具体的通信协议来实现的。要实现有条不紊地进行数据通信，通信双方就必须事先约定好相应的规则，就像我们平常与人合作做一件事时事先谈好合作的方式和注意事项一样。当然，计算机网络中的数据通信规则不是通信双方具体签定的，而是由双方所采用的网络通信协议来完成的。

在利用网络通信协议进行计算机网络通信时主要涉及以下 3 个方面的问题：一是指出本协议要实现的网络服务；二是规定如何实现这些指定的网络服务；三是如何与对方实现协同工作。这 3 个方面也就对应了计算机网络通信协议的 3 个基本要素：语义、语法和同步。下面分别解释它们各自的功能。

1. 语义

“语义”可以理解为“语意”，是用来解决“做什么”的问题，也就是描述该通信协议具体用来实现什么功能。如我们平常工作和生活中所签的协议中的标题或者主题一样，让人家一看就知道该协议是用来做什么的。

双方在对等层次进行通信时，首先就要确定双方所使用的协议的“语义”是否一样，也就是完成的功能是否一样。如一方用户使用的是 L2TP 协议进行 VPN 通信（也就相当于他说“我要进行 L2TP VPN 连接”），而另一方却使用 PPTP 协议来进行 VPN（也就相当于他说“我要进行 PPTP VPN 连接”），目的或者方式都不一样，自然是不能成功协商的。

在通信协议的“语义”部分还包括通信协议的版本，就像我们平常所签的协议也有版本（主要是根据签协议的时间先后来判断）一样。不同的版本，所能实现的功能以及实现的方式都可能存在大的差别，当然高版本的通信协议通常是向下兼容低版本的。

2. 语法

“语法”用来规定通信时的信息格式，包括数据及控制信息的格式、编码及信号电平等，规定实现某项服务功能双方所必须遵守的一些基本要求，是用来解决“如何做”的问题。如我们平时所签的协议中规定的具体条款，规定如何确保达到最终目标一样。如果双方进行网络通信时所用的通信协议一样，或者只要双方所使用的通信协议的“语义”部分是一样的，也就相当于双方的目标是一致的。接下来的问题就是如何来实现双方共同的目标，完成相应的网络服务。

通信协议也是软件，也是由一些计算机程序语言来开发的，所以协议的具体语法则依据所采用的程序开发语言的语法规则来进行。尽管有时，双方所使用的通信协议名称并不一样，但只要其语

义部分相同，且双方都能识别对方的语法规则，则仍然是可以进行相互通信的。就相当于我们与别人对话时，如果双方都懂几种语言，尽管母语不一样，但是都能够听懂对方在说什么，明白对方要做什么，仍然可以进行交流，共同完成某项任务。

3. 同步

“同步”是用来解决“做的次序”的问题，也就是通信双方要完成某项网络服务必须依据什么样的流程、使用什么样的速率来进行。在网络通信中这称为“握手”。通信协议是用来实现某项网络服务的，但在通信双方必须保持一定的程序执行步骤，就像打电话时的一问一答模式一样，否则一方的请求得不到对方的应答时就会出现程序错误，最终导致通信失败。

同时，因为程序执行的各个步骤之间是有相互依赖性的，有时要依据上一步骤的用户选择来做出下一步的选择的。再一个，还要考虑通信双方“说话”的速度问题，如因某些原因一方的程序执行效率比较高（如网络带宽高、设备性能好），另一方的程序执行效率低（如网络带宽低、设备性能差），这时就得双方协商好如何保持一个双方都可接受的执行速率，这样才能做到步调一致，以防出现差错。其实这就是我们在本书后面经常要提到的“流量控制”和“拥塞控制”功能。如果不协商好传输速率的话，一方传输完数据半天对方还在接收，这时可能会因为传输延时等问题而出现数据丢失，导致对方接收到的数据不完整。就像我们平时打电话一样，如果双方都讲个不停，那么谁也听不清对方说什么，或者一方说话太快，对方根本听不清。

3.8 课后自测题

为了巩固本章所学的内容，请认真做下面的自测题，参考答案见附录。

3.8.1 填空题

1. 计算机网络体系结构诞生的雏形是在_____中出现的“资源子网”与“通信子网”的划分，在 OSI/RM 体系结构诞生之前，与它最为相似的体系结构是由_____公司开发的_____体系结构。

2. OSI/RM 分为七层，其中资源子网包括_____、_____、_____和_____，最低层是_____，最复杂、协议最多的是_____，提供数据压缩和解密的是_____。

3. 最初的 TCP/IP 体系结构中从低到高划分为_____、_____、_____和_____四层，其中最高层包括了 OSI/RM 的_____、_____和_____这三层。

4. 在局域网体系结构中，数据链路层分为_____和_____两个子层，IEEE 802.3 标准工作在_____，IEEE 802.1q 工作在_____。

5. 在计算机网络通信中，各层之间的通信接口称为_____，计算机网络通信协议的 3 个基本要素是_____、_____和_____。

3.8.2 选择题（可多选）

1. 在 OSI/RM 中第一个提供端到端服务的层次是（　　）。

A. 数据链路层　　B. 网络层　　C. 传输层　　D. 应用层

2. 在 OSI/RM 中用来构建网络通信平台的层次不包括（　　）。

A. 物理层　　B. 数据链路层　　C. 网络层

D. 传输层　　E. 会话层

3. 在 OSI/RM 中实现局域网内部连接和通信的层次包括（　　）。

A. 物理层　　B. 数据链路层　　C. 网络层

D. 传输层　　E. 会话层

4. 在 OSI/RM 中实现跨网连接的层次是（　　）。

A. 物理层　　B. 数据链路层　　C. 网络层

D. 传输层　　E. 会话层

5. 下列功能不属于表示层提供的是（　　）。

A. 交互管理　　B. 数据透明传输　　C. 数据加密

D. 数据压缩　　E. 数据格式转换

6. 在 OSI/RM 中，网络层的主要功能是（　　）。

A. 组织两个会话进程之间的通信并管理数据的交换

B. 数据格式变换、数据加密与解密、数据压缩与恢复

C. 路由选择、拥塞控制与网络互连

D. 确定进程之间通信的性质以满足用户的需要

7. OSI 参考模型的物理层中没有定义（　　）。

A. 硬件地址　　B. 位传输　　C. 电平　　D. 物理接口

8. OSI/RM 网络层的数据单元是（　　）。

A. 数据段　　B. 数据包　　C. 数据分组　　D. 比特

9. 在 OSI/RM 中不会进行协议封装的是（　　）。

A. 物理层　　B. 数据链路层　　C. 网络层

D. 传输层　　E. 应用层

10. 在 OSI/RM 中只有（　　）可以对数据封装协议尾。

A. 物理层　　B. 数据链路层　　C. 网络层

D. 传输层　　E. 应用层

11. 在 OSI/RM 中可以实现差错控制和流量控制的是（　　）。

A. 物理层　　B. 数据链路层　　C. 网络层

D. 传输层　　E. 应用层

12. 传输层是通过（　　）标识不同应用进程的。

A．物理地址　　B．端口号　　C．IP 地址　　D．逻辑地址

13．我们在连接 WLAN 网络时所用的 IEEE 802.11a/b/g/n 标准工作在（　　）。

A．物理层　　B．数据链路层　　C．MAC 子层　　D．LLC 子层

14．IEEE 802.11i 工作在（　　）。

A．物理层　　B．MAC 子层　　C．LLC 子层　　D．应用层

15．在网络通信协议的三要素中用来进行通信参数协商的是（　　）。

A．语义　　B．语意　　C．语法　　D．同步

16．在网络通信协议的三要素中（　　）用来解决“如何做”的问题。

A．语义　　B．语意　　C．语法　　D．同步

17．正确描述了数据封装过程的是（　　）。

A．数据段→数据包→数据帧→数据流→数据

B．数据流→数据段→数据包→数据帧→数据

C．数据→数据包→数据段→数据帧→数据流

D．数据→数据段→数据包→数据帧→数据流

18．从网络层发出的数据包，在接收端最终将在（　　）被处理。

A．应用层　　B．传输层　　C．网络层　　D．物理层

19．TCP/IP 体系结构把 OSI/RM 中的（　　）合并到应用层。

A．传输层和表示层　　B．传输层和会话层

C．会话层和表示层　　D．表示层和应用层

20．下面关于局域网用户数据通信的说法正确的是（　　）。

A．局域网内的用户数据通信只需要物理层和数据链路层

B．局域网内传输的数据单位是数据帧，是通过 MAC 地址进行寻址的

C．局域网内的用户数据直接在数据链路层封装成数据帧

D．局域网内的用户数据通信也必须使用体系结构中的全部层次

21．TCP/IP 体系结构中的（　　）规范了数据传输速率、最大传输距离等参数。

A．物理层　　B．数据链路层　　C．传输层　　D．应用层

22．下面关于“对等通信原理”的说法正确的是（　　）。

A．数据在传输过程中每经过一个设备都必须在同一个层次被转发

B．对等通信只针对源端和目的端，不涉及中间设备

C．本端设备必须与相邻对端设备工作在同一层次

D．所有数据都必须从本端应用层发送，也必须在目的端的应用层接收

23．下面关于协议封装和解封装的说法不正确的是（　　）。

A．在发送端数据经过体系结构的每一层都要经过对应层协议的封装

B．协议封装的目的是为了带上本层协议的地址信息

C．协议解封装是为了在数据向上层传递时使上层能识别本层协议的地址信息

D．协议封装和解封装都只在源端和目的端设备上进行

24．下面关于数据传输的说法不正确的是（　　）。

A．无论哪层发送的数据都必须经过传输介质以比特流方式传输到对方设备上

B．每个接口在发送数据前都必须重新对数据进行协议封装

C．在同一个局域网内，数据中的 MAC 地址信息是不会改变的

D．数据在经过不同网络时，数据中的 MA 地址信息会改变，但 IP 地址信息不会变

25．下面关于计算机网络体系分层结构的说法正确的是（　　）。

A．每层都是完全独立的，相互之间没有依存关系

B．上层功能的实现必须建立在下层功能完成的基础上

C．所有计算机网络功能都必须依靠体系结构中的所有层次来共同完成

D．SAP 位于上层与下层之间，负责相邻层次之间服务的相互调用

4

计算机网络通信基础

在正式介绍计算机网络体系结构中各层的主要功能及相关技术原理前，本章先来对计算机网络通信的一些基础知识进行介绍。

本章所介绍的内容非常重要，但也非常难懂，因为都涉及一些比较复杂的技术原理，其中主要包括通信系统基本模型、基带传输系统、频带传输系统、各种基于数据传输的分类、数据传输速率与带宽、各种基带数字信号的编码原理，以及各种频带传输系统中所使用的基带数字信号调制与解调原理等。

本章重点与难点：

- 基带传输与频带传输的区别
- 传输速率与信道带宽的计算
- 奈奎斯特准则、香农公式和采样定律
- 归零码与非归零码的区别
- AMI、CMI、HDB3、曼彻斯特码和差分曼彻斯特码编码原理
- 调制与解调工作原理的理解
- 各种 2ASK、2FSK、2PSK 的调制与解调方法工作原理

4.1 数据通信系统基本模型

数据通信是计算机与计算机或计算机与终端之间的通信。它传送数据的目的不仅是为了交换数据，更主要是为了利用计算机来处理数据。可以说它是将快速传输数据的通信技术和数据处理、加

工及存储的计算机技术相结合，从而给用户提供及时正确的数据。

在计算机网络数据通信模型中，涉及两个实体和一个通信信道，它们是源系统、目的系统和传输系统。其中源系统又包括源站点和发送器，目的系统又包括目的站点和接收器。如图 4-1 显示了计算机网络数据通信系统的基本模型组成，同时以一个最简单、最典型的 Modem 通过 PSTN（公共交换电话网络）拨号远程计算机访问为例解释了这几个组成部分的基本作用。

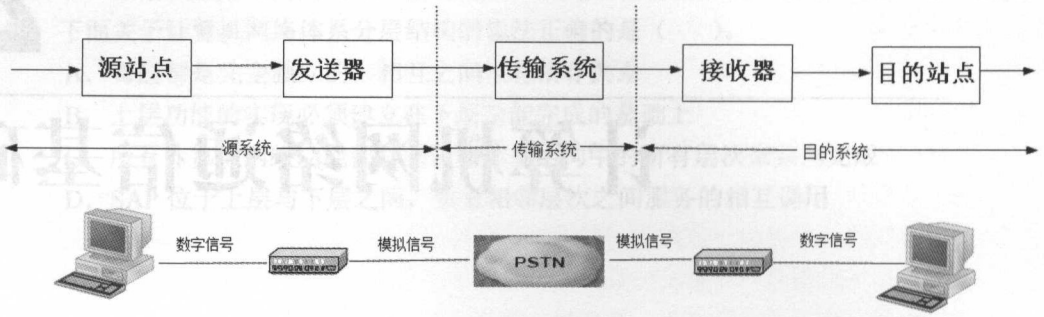


图 4-1　数据通信系统基本模型

1. 源系统

“源系统”就是发送信号的一端，即发送方。它包括以下两个必需的部分：

- 源站点：产生要传输的数据的计算机或服务器等终端设备。
- 发送器：对要传送的数据进行编码或者调制的设备，如各种调制解调器、收发器和计算机网卡。

2. 传输系统

“传输系统”是计算机网络上的数据传输通道，除了“源系统”和“目的系统”外，网络中的其他部分都属于传输系统，包括通过物理层传输介质和相关协议建立的物理链路以及结点设备（如网络中的交换机、路由器等设备）。这部分是整个计算机网络中最复杂的一部分，不仅网络结构千差万别，而且所使用的网络通信协议和最终所能实现的网络功能也是各种各样的。

3. 目的系统

“目的系统”就是接收发送端所发送的信号的一端，即接收方。与“源系统”一样，它也包括以下两个必需的部分：

- 目的站点：从接收器获取从发送端发来的数据的计算机或服务器等。
- 接收器：接收从源站点发来的信息并把它们转换为能被目的设备识别和处理的信息的设备，主要也是各种调制解调器、收发器和计算机网卡。

4.2　信号传输系统分类

在计算机网络中，信号的传输主要有两种方式：一种是信号无需经过调制直接进行的传输，称

为“基带传输”；另一种是在传输前必须对信号进行调制，然后再进行的传输，称为“频带传输”。另外还有一种称为“宽带传输”，本节将具体介绍这3种传输方式所对应的信号传输系统。

4.2.1 基带与频带

要理解什么是“基带传输”和“频带传输”，首先要理解的就是什么是“基带”（Baseband）和“频带”（Frequencyband）。

“基带”是指由信源发出的，没有经过调制（即没有进行频谱搬移和变换）的原始电信号所固有的频带（频率带宽），也称为“基本频带”。而“频带”是指对基带信号调制后所占用的频率带宽。有关调制技术和原理将在本章后面具体介绍。

1. 基带信号和频带信号

“基带信号”是指由信源发出，没有经过调制的原始电信号（频带属性没有改变）。基带信号的特点是频率较低，信号频谱通常是从零频率的直流成分开始的。

相反，“频带信号”是经过调制后的基带信号。很显然，频带信号是把基带信号的频带属性进行了改变，被限制在一个特定的频带中的基带信号。如果一个信号只包含了一种频率的交流成分或者有限几种频率的交流成分，那么它上面肯定是频带信号。

【说明】以上所说的“基带信号”和“频带信号”不仅适用于数字信号，同样适用于模拟信号，所以基带（或频带）信号可分为数字基带（或频带）信号和模拟基带（或频带）信号，具体由其信源类型确定。因为数字信号是离散的脉冲信号，所以基带信号往往包含丰富的低频分量甚至直流分量，不适合长距离传输。

2. 基带传输和频带传输

通过对以上“基带信号”和“频带信号”定义的了解，我们很容易知道，在信道中直接传输不经过调制的基带信号的方式称为“基带传输”，而在信道中传输经过调制后的频带信号的方式称为“频带传输”。

在基带传输中由于不调制，所以整个信道只传输一种信号，通信信道利用率低。但为了没有经过调制的信号能在传输信道中有效地传输，需要在信源端对基带数字信号进行编码，然后再在信宿端对编码的信号进行解码恢复。这分别是由对应传输系统的编码器和解码器完成的，具体将在下节介绍。

由于基带信号频率一般比较低，因此不适合远距离传输，在计算机局域网（如令牌网、以太网）内的数据传输大都采用基带传输类型。而频带传输中对基带信号所进行的调制通常是用一个高频信号携带上低频的基带信号进行传输，所以可使最终的信号频率得到提高，更适合远距离传输。所以在广域网远程网络连接（如通过各种 Modem 拨号连接）中基本上都是采用频带传输方式。

4.2.2 数字基带传输系统和数字频带传输系统

在实现信号的基带传输或频带传输时都必须有一整套组件来帮助完成，本节分别介绍基带传输

系统和频带传输系统的构成。因为在计算机网络通信中主要是数字信号，所以我们这里仅针对数字基带传输系统和数字频带传输系统进行介绍。

1. 数字基带传输系统

如图 4-2 所示是数字基带传输系统的基本组成，主要由编码器、发送端滤波器、传输信道、接收端滤波器、抽样判决器和解码器组成。此外为了保证系统可靠有序地工作，还应有位同步系统。

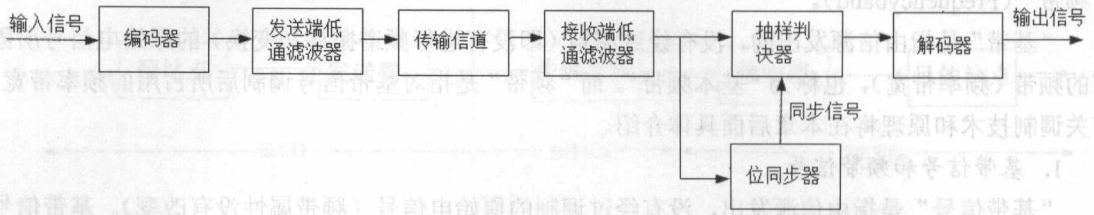


图 4-2　数字基带传输系统的基本组成

（1）编码器。

不同传输介质所支持的信号编码类型不同，而原始数字基带信号通常是单极性不归零码（NRZ），所以在基带传输中，在信号真正传输到传输介质前要将信源输出的信号编码类型转变为适合于对应信道传输的码型。有关数字基带信号的编码类型将在本章后面专门介绍。

（2）发送端低通滤波器。

为了使经过编码的基带信号的频带特性与传输介质的频带特性相符，去掉较高频率的无用信号和干扰信号，经过编码的基带信号在传输介质上发送之前需要通过低通滤波器进行过滤，使仅符合低通滤波器频带特性的信号通过。

这种变换主要是通过波形变换来实现的，其目的是使信号波形与信道匹配，便于传输，减小码间串扰，以有利于在接收端的同步提取和抽样判决工作进行。

（3）传输信道。

经过低通滤波器过滤的基带信号就可以正式在传输介质上的传输信道中传输了。

（4）接收端低通滤波器。

信号在传输过程中，特别是长距离传输过程中，信号会被干扰，所以在信号到达接收端时先要进行一次低通滤波，滤除基带（基本频带信号）以外的频带信号，并对信道特性进行均衡，使输出的基带波形无码间串扰，以有利于接收端的抽样判决工作进行。

（5）抽样判决器。

尽管前面对信号进行了低通滤波，但在低通频带中仍可能存在干扰信号，而且这些干扰信号是连续的。为了尽可能还原离散特性的原始数字基带信号，就需要对信号进行离散抽样，使输出的信号具有数字基带信号的离散特性。它是与下面的“位同步器”协同工作的。

（6）位同步器。

“抽样判决器”只负责对信号进行抽样，至于在信号的什么位置、按什么频次进行抽样，这是

由“位同步器”负责完成的。位同步器通过生成与原始基带信号同步的同步信号控制抽样判决器比较精准地提取信号，并且还可以控制抽样的频率，使得最终抽样的信号保持与原始数字基带信号同样的离散和电平属性。

（7）解码器。

对抽样判决器输出的信号进行解码，最终还原出有用的原始基带信号，并根据需要求对得到的信号编码类型进行转换，使其符合接收端的码型要求。

2. 数字频带传输系统

要进行信号的频带传输，也必须有一整套频带传输系统。在前面介绍数字基带传输系统中，为使数字基带信号能够在信道中传输，要求信道应具有低通（也就是所有低于某个频率的信号均可通过，高于某个频率的信号全部滤除）形式的传输特性。而大多数实际信道具有带通（也就是只有频率在某个范围内的信号才能通过，其他的均将滤除）传输特性，这样一来基带信号不能在这种带通传输特性的信道中传输，必须进行调制。

调制就是把一个调制信号放在另一个频率更高的载波信号上，使在信道中传输的信号具有两种信号波的双重特性，然后通过调制信号的某些特性（如振幅、相位）影响载波信号的相应特性的一种技术。具体将在本章后面介绍。

数字频带传输系统与数字基带传输系统的组成基本类似，不同的只是在频带传输系统的接收端要采用“调制器”替代基带传输系统中的“编码器”，采用“发送端带通滤波器”替代基带传输系统中的“发送端带低通滤波器”，在频带传输系统的接收端要采用“解调器”替代基带传输系统中的“解码器”，采用“接收端带通滤波器”替代基带传输系统中的“接收端带低通滤波器”，如图 4-3 所示。

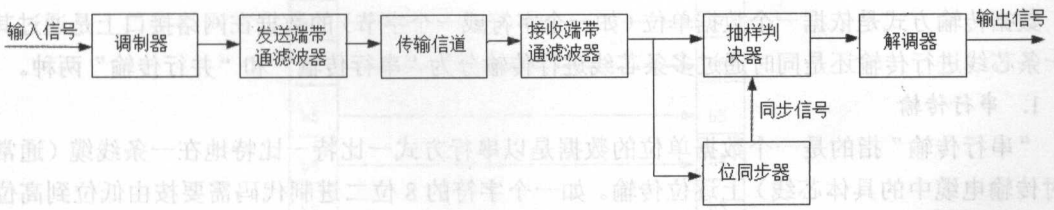


图 4-3 数字频带传输系统的基本组成

“调制器”的作用是对原始的数字基带信号进行调制，最终使信号可在以具有某个带通属性的传输信道中传输。在调制过程中会引入具有高频特性的载波信号，所以信号在信道传输过程中用于信号滤波的就不再是低通滤波器了，而必须是符合对应传输介质频带特性的带通滤波器。同时，由于对原始基带数字信号进行调制时会加上载波信号，在传输信道中传输的自然就不只是有用的原始基带信号了，所以在接收端就必须用到“解调器”来去掉所接收的信号中附加的载波信号，还原有用的基带信号。

4.2.3 宽带传输和宽带传输系统

除了前面介绍的“基带传输”和“频带传输”外，在传输方式上还有一种称为“宽带传输”的方式。这里所说的“宽带”也是基于信号频率带宽而言的。它就是采用比音频（4kHz）更宽的频带，可以包括大部分电磁波频谱，是相对前面所说的“频带”而言的，因为“频带”只具有有限的频率宽度。使用这种较宽频带进行的数据传输就称为“宽带传输”。

“宽带传输”其实与“频带传输”具有基本相同的特点，也是需要对信源发出的原始信号进行调制，所以在整个传输系统的组成上基本一样，当然在滤波器方面，这里要采用“宽带滤波器”。另外，在宽带传输中会将一个信道分成多个子信道，这样就可以在一个物理信道上分别传输像音频、视频和数据等多路信号。

最常见的宽带传输就是我们天天用的有线电视，在一根同轴电缆中可以传输那么多电视频道信号，另外无线广播也是采用宽带传输方式的。

4.3 数据传输分类

在进行数据传输前，还有几个方面要事先确定好，其中包括一个单位数据是只通过一条芯线传输还是由多条芯线同时传输（数据传输方式）、数据传输时是否要求发送端和接收端采用统一的时钟参考（数据传输模式）、同一时刻同一条芯线上是否允许同时进行两个方向的数据传输（数据通信模式）等，本节将分别进行具体介绍。

4.3.1 数据传输方式

数据传输方式是依据一个数据单位（如一个字符或一个字节）的数据在网络接口上是通过其中的一条芯线进行传输还是同时通过多条芯线进行传输分为“串行传输”和“并行传输”两种。

1. 串行传输

“串行传输”指的是一个数据单位的数据是以串行方式一比特一比特地在一条线缆（通常是针对传输电缆中的具体芯线）上逐位传输。如一个字符的 8 位二进制代码需要按由低位到高位的顺序依次传输，如图 4-4 所示。等第一个字符的最高位传输完后，再传输第二个字符的最低位，依此类推。

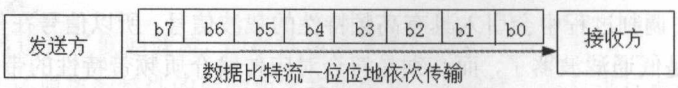


图 4-4 串行数据传输原理示例

很显然，串行传输方式对于一个单位数据的传输来说理论上只需要一条传输信道（即一条芯线）即可，而且易于实现，成本也可以很低。但这种传输方式的传输速率比较慢，为此通常在进行串行

传输类网络接口设计时也采用多芯线方式，这样每条芯线可以分别、同时传输一个单位数据，可以成倍提高传输速率。

在计算机中采用串行数据传输方式的接口比较多，如常用的USB接口、SATA（Serial ATA，串行ATA）磁盘接口、SAS（Serial Attached SCSI，串行连接的SCSI）磁盘接口等。在计算机网络接口中采用串行传输方式的就是里面包含不同数量芯线的各种串口，如RS-232C接口（即Serial接口）通常是包含8根用于数据传输的芯线，RS-422接口是包含4根用于数据传输的芯线。我们常用的以太网双绞线中的四对芯线也是采用串行传输方式，每一对分别用于传输不同数据单位的数据。

传输速率虽然不会给串行传输方式带来问题，但还是存在一个问题，那就是接收端如何与发送端保持数据单位的同步，使得接收端能够区分不同数据单位的比特位边界，就涉及到在下节将要介绍的“异步传输模式”和“同步传输模式”。

2. *并行传输*

“并行传输”指的是一个单位的数据（如一个字节或者一个字符）可以通过传输电缆中的多条芯线进行传输，也就是网络接口中的多条芯线是同时负责一个单位数据的传输的，而不是像串行传输方式那样每条芯线分别传输不同的单位数据。

如图4-5所示是在一个包含8根芯线的电缆上采用并行传输方式传输一个字符（8个比特）的示意图。由于同一时刻多条芯线上同时传送了整个数据单位的所有比特位，两次传输间要有一定的时间间隔，所以接收端与发送端之间不存在数据单位同步的问题，这是并行传输的一个主要优点。

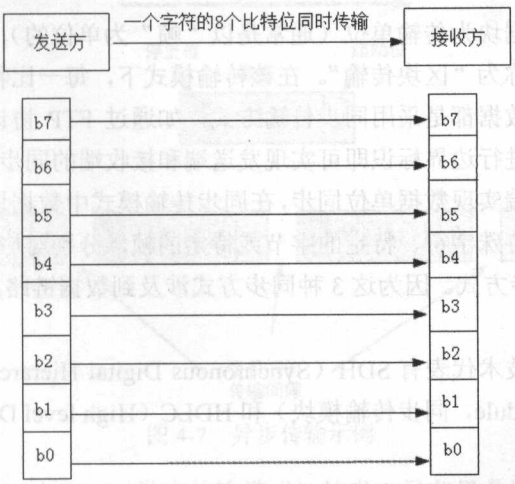


图4-5 并行数据传输原理示例

并行传输方式的缺点也非常明显，首先在一条电缆中必须有多条并行信道（即要有多条芯线），则通常需要在一条电缆中至少包括8条芯线（因为一个字节包括8比特），这在设施成本上将带来比较大的负担，特别是远距离传输时；其次，并行传输通常要求每个信道在传输完一个比特后要停

下来，以便接收端可以依据这个时间间隙进行数据单位同步，这大大影响了数据的传输效率。基于以上两方面的原因，并行传输方式主要适用于计算机和其他高速数据系统的近距离传输，在计算机网络中的应用比较少见。

采用并行传输方式的典型代表有计算机上连接打印机的并口、磁盘 ATA（或 IDE）接口，计算机主板上 CPU 与其他芯片的数据传输也是采用并行传输方式。

4.3.2 数据传输模式

在上节中介绍到，计算机网络中的数据传输一般不是采用并行传输方式而是串行传输方式。这要解决串行传输方式中的一个最主要的问题，即数据单位同步的问题。发送端一位一位地把数据通过传输介质发往接收端，接收端必须能识别出一个个数据单位的起始和结束，而且必须知道每一位的持续时间。只有这样，接收端才能从传输线路上正确地从收到的比特流中正确组装、还原出一个个数据单位。

为了解决串行传输方式中的数据单位同步问题，在数据传输模式上又采取了“同步传输”（Synchronous Transmission）和“异步传输”（Asynchronous Transmission）两种模式。

1. 同步传输

“同步传输”中关键是要理解“同步”的含义。“同步”是指通信双方在传输过程中是同步进行的（也就是接收端与发送端同时开始工作，并且接收端按数据的发送顺序进行接收），同步的依据就是双方有相同的时钟参考，能同时开始数据的发送和接收。通常这个时钟参考是由同步时钟芯线或同一个时钟源提供。

同步传输是一种以数据块为传输单位（通常是以“帧”为单位的），以相同的时钟参考所进行的数据传输模式，因此也称为“区块传输”。在该传输模式下，每一比特数据的持续传输时间都是相等的。大多数非字符类数据都是采用同步传输方式，如通过 FTP 协议进行各种软件文件传输，这时只需要对一个数据块进行边界标识即可实现发送端和接收端的同步。

为了接收端能与发送端实现数据单位同步，在同步传输模式中数据块的开始和结尾部分都加上了一个用于数据块同步的特殊字符、特定的字节或特定的帧，分别对应“面向字符”“面向比特”和“面向字节”这 3 种同步方式。因为这 3 种同步方式涉及到数据链路层的“成帧”功能，所以具体将在第 6 章中介绍。

采用同步传输模式的技术代表有 SDH（Synchronous Digital Hierarchy，同步数字系列）、STM（Synchronous Transfer Module，同步传输模块）和 HDLC（High level Data Link Control，高级数据链路控制）等。

与下面将要介绍的“异步传输”方式比较，“同步传输”在技术上较复杂，但因为同步传输模式是基于以帧为单位的数据块（不是基于字符），不需要对每个字符加单独的“起始”和“停止”比特，接收方不必对每个字符进行开始和停止的操作，只需要检测到帧同步字符即可，因此传输效率高，常用于较高速的数据传输。

2. 异步传输

“异步传输”的关键也是“异步”这两个字，是指通信双方没有相同的时钟参考（也就是发送端和接收端不需要同时开始工作），但双方在数据传输速率上是同步的。即每个字符之间的传输是异步的，但在一个字符内的每一位的传输还是同步的。如我们的键盘输入都是采用异步传输模式，每个字符是单独传输的。

异步传输是以字符为单位进行数据传输的。在发送每一字符代码的前面均加上一个“起始位”信号（1 比特，极性为 0），用以标记一个字符的开始；在一个字符代码的最后也加上一个“停止位”信号（1 或 2 比特，极性为 1），用以标记一个字符的结束。数据位占 5～8 位，具体取决于数据所采用的字符集，如电报码字符为 5 位，ASCII 码字符为 7 位。

异步传输模式下，每个字符在传输时的数据结构如图 4-6 所示。图 4-7 所示是一个异步传输模式的数据传输示例，假设所传输的字符二进制代码为 11111011，经过加装“起始位”和“停止位”后为 **1**1111110**11**0。

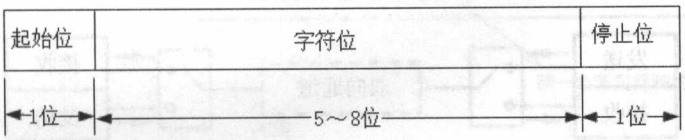


图 4-6 异步传输中的字符传输格式

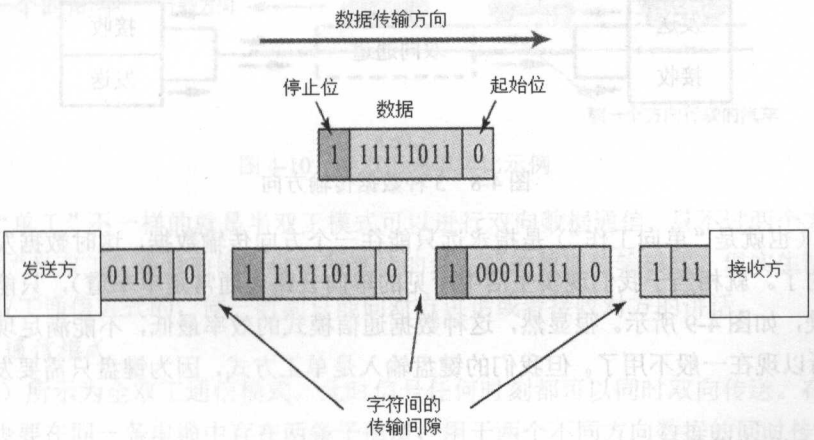


图 4-7 异步传输示例

异步传输的典型代表就是 ATM（异步传输模式）技术，目前仍是城域网和广域网中主流的交换技术之一。早期还有一种 B-ISDN（宽带 ISDN）技术，也是异步传输技术的代表，通过 FTP 协议以 ASCII 方式进行的数据传输也是异步传输模式。

异步传输的优点是字符同步实现简单，收发双方的时钟信号不需要严格同步；缺点是该传输模式仅适用于字符数据传输，而且要对每一个字符都加上“起始位”和“停止位”，码元利用率低，

最终使得有效传输率降低。如 ASCII 编码字符中本身有一位用于奇偶校验，现在再增加起始、停止各一位，就相当于传输一个字符要用 10 位，而真正有效的数据才 3 位，传输效率只有 70%。

4.3.3 数据通信模式

在数据通信中按照通信双方的通信模式是根据同一时刻在同一条或同一对传输介质芯线是否允许进行双向数据传输来划分的，包括“单工模式”“半双工模式”和“全双工模式”3 种。

1. 单工通信模式

图 4-8（a）所示为单工通信模式。在单工通信中，数据只能向一个方向传输，任何时候都不能改变数据的传送方向。

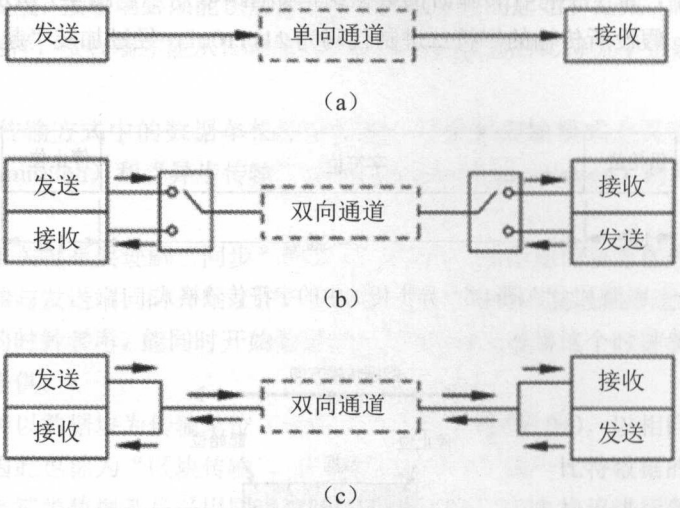


图 4-8　3 种数据传输方向

“单工”（也就是“单向工作”）是指永远只能往一个方向传输数据，这时数据发送方和接收方就都是固定的了。就相当于我们现实生活中所见的单向公路（通常是单车道），只能允许汽车朝同一个方向行驶，如图 4-9 所示。很显然，这种数据通信模式的效率最低，不能满足现在我们的数据通信需求，所以现在一般不用了。但我们的键盘输入是单工方式，因为键盘只需要发送数据，不需要接收数据。

2. 半双工通信模式

图 4-8（b）所示为半双工通信模式，从中可以看出它的数据传输方向是受一个类似于开关来控制的。在半双工通信模式中，数据是可以双向传输的，但必须是交替进行，即同一时刻只能向一个方向传输，在不同时刻可以进行另一个方向的数据传输。

半双工通信模式就相当于可以双向行驶的单车道，同一时刻只允许一个方向的汽车通过，只有当另一个方向没有车辆过来的时候才能从自己一方发车，如图 4-10 所示。

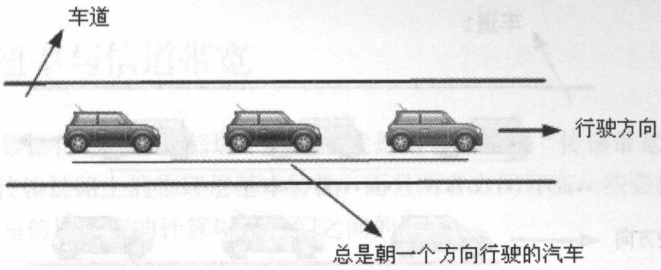


图 4-9 单工通信类比示例

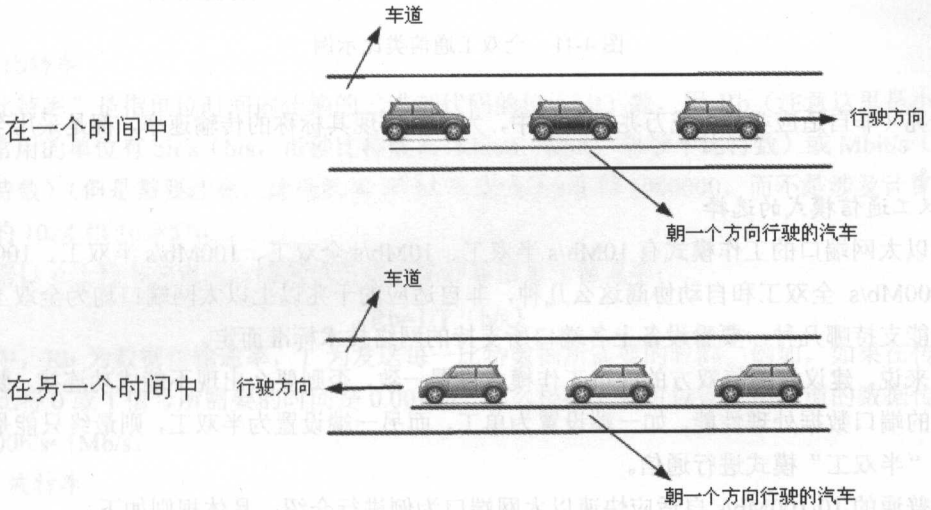


图 4-10 半双工通信类比示例

与前面的“单工”不一样的就是半双工模式可以进行双向数据通信，只不过两个方向不能同时进行数据传输。“半双工”是目前网络设备所支持的最主要的数据传输模式。现实生活中，一般的对讲机都是半双工通信方式的，同一时间只能向对方讲话或者接收对方的讲话。

3. 全双工通信模式

图 4-8（c）所示为全双工通信模式。此时信号任何时刻都可以同时双向传送。在全双工通信模式中，它至少要在同一条电缆中存在两条子信道，用于两个不同方向数据的同时传输。

全双工通信模式就相当于我们现实生活中的“双向双车道”。在这样的公路中，两个方向的车子都可以同时通过，如图 4-11 所示。很明显，这样可以提高车子通过的效率。对于数据通信来说，全双工线路的传输速率就提高了数据传输速率，最高可以达到带宽的两倍。如 100Mb/s 的全双工线路，最终的数据传输速率最高可达 200Mb/s。这也就是为什么 4 个 100Mb/s 端口聚合而成的以太网通道最高可以达到 800Mb/s 的数据传输速率的原因了。

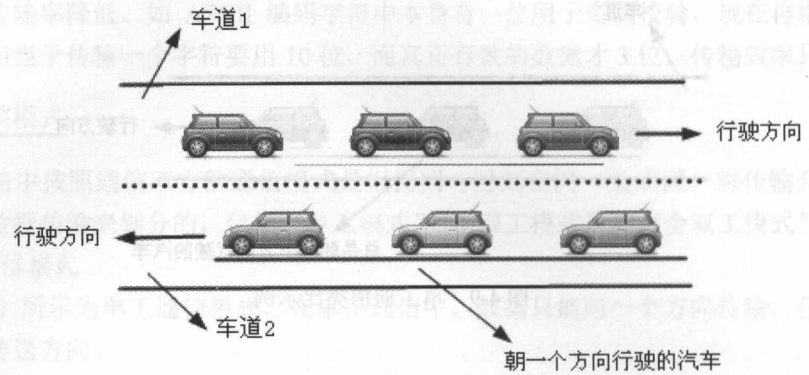


图 4-11　全双工通信类比示例

在千兆（非自适应类型）或万兆以太网中，为了能实现其标称的传输速率，都是采用全双工通信方式的。

4. 双工通信模式的选择

一个以太网端口的工作模式有 10Mb/s 半双工、10Mb/s 全双工、100Mb/s 半双工、100Mb/s 全双工、1000Mb/s 全双工和自动协商这么几种，非自适应的千兆以上以太网端口均为全双工模式。当然最终能支持哪几种，要看设备上各端口所支持的网络技术标准而定。

一般来说，建议将通信双方的端口工作模式设置一致，否则要么出现不能成功连接，要么会降低另一端的端口数据处理性能。如一端设置为单工，而另一端设置为半双工，则最终只能是选择最低要求的“半双工”模式进行通信。

现以普通的 10/100Mb/s 自适应快速以太网端口为例进行介绍，具体规则如下：

- 如果一端配置为半双工模式（无论是 10Mb/s 还是 100Mb/s），另外一端配置为自协商模式，自协商的那一端也将只能工作在半双工模式。
- 如果一端配置为全双工模式，另一端配置为半双工模式，Ping 是没有问题的，流量小的时候也没有任何问题，流量达到约 15%以上时就会出现冲突、错包，最终影响了工作性能。
- 对于两端均配置为自动协商模式，最后协商成的结果是“两端都支持的工作模式中优先级最高的那一类”，通常为半双工模式优先。
- 如果 A 端配置为自动协商，B 端配置为 100Mb/s 全双工，则在 A 协商的结果为 100Mb/s 半双工后，再强制将 B 改为 10Mb/s 全双工，则 A 端也会马上向下协商到 10Mb/s 半双工。
- 如果 A 端配置为自动协商，B 端配置为 10Mb/s 全双工，则在 A 协商为 10Mb/s 半双工后，再强制将 B 改为 100Mb/s 全双工，会出现协商不成功连接不上的现象。这个时候，如果插拔一下网线，又会重新协商在 100Mb/s 半双工。

4.4 数据传输速率与信道带宽

在数据通信中，数据传输速率与信道带宽（也就是通常所说的“传输带宽”）是两个非常重要的概念。它们定义了传输链路上的物理层基本特性，而且两者之间存在一些必然的联系，本节将详细介绍数据传输速率与信道带宽的计算以及它们之间的关系。

4.4.1 传输速率与信道带宽的基本概念

“数据传输率”是指在一条信道中单位时间内传输的信息量，可用“比特率”或“波特率”来表示。

1. 比特率

“比特率”是指单位时间内传输的二进制代码的位（bit）数，用 Rb（注意这里是小写的 b）表示。常用的单位有 bit/s（b/s，每秒比特数）、Kbit/s（Kb/s，每秒千比特数）或 Mbit/s（Mb/s，每秒兆比特数）（但是需要注意，此处的 K 和 M 分别为 1000 和 1000000，而不是涉及计算机存储器容量时的 1024 和 1048576）。

根据以上比特率的定义，可用以下公式来计算信息传输速率：

$$Rb=1/T\ (b/s) \qquad ①$$

其中，Rb 为数据传输速率，T 为发送每一比特数据所需要的时间。例如，如果在传输信道上发送 1 比特 0 或 1 信号所需要的时间是 0.001ms，那么根据公式可以计算出信道的数据传输速率为 1 000 000b/s=1Mb/s。

2. 波特率

“波特率”（Baud rate）是指一个基带数字信号在被调制后基带数字信号对载波的调制速率，亦即单位时间内载波参数（如频率、相位等）变化的次数，单位为 B。如果是没有经过调制的二进制数字信号，“波特率”与“比特率”是相等的，所以“波特率”通常是针对经过调制后的数字信号或者模拟信号的传输速率。

波特率的单位为 Baud，用 RB（注意这里是大写的 B）表示。一个数字脉冲或者一个二进制位或多进制位称为一个码元。波特率可以理解为在单位时间内传输的码元数。若 1 秒内传输 2400 个码元，则波特率为 2400B。

3. “比特率”与“波特率”的关系

对于数字数据，“比特率”与“波特率”的关系与数据所采用的进制有关。因为每个码元或符号通常都含有一定 bit 数的信息量，所以“比特率”（Rb）与“波特率”（RB）有如下等式关系：

$$Rb=RB\times \log_2 M\ (b/s) \qquad ②$$

式中的 M 为符号的进制数，如二进制、八进制、十六进制。例如码元速率为 1200B，采用八进制（M=8）时，则可以计算出对应的数据传输速率 $Rb=1200\times \log_2 8=3600b/s$；如果采用的是二进制（M=2），则数据传输速率为 $Rb=1200\times \log_2 2=1200b/s$，可见二进制的波特率和比特率在数量上是

相等的。

在采用了数字调制技术的数字通信中，比特率与波特率的关系还与所采用的调制技术有关。因为具体的调制技术将在本章后面有专门介绍，所以在此仅给出它们的关系等式，也与公式②一样。不过，此时的 M 代表的是调制的相位数。这样一来可以得出，两相调制（单个调制状态对应 1 个二进制位）的比特率等于波特率；四相调制（单个调制状态对应 2 个二进制位）的比特率为波特率的两倍；八相调制（单个调制状态对应 3 个二进制位）的比特率为波特率的 3 倍；依此类推。

4. 传输带宽

“带宽”是指信道中每秒钟传输的最大信息量，也就是一个信道在理想状态下可达到的最大数据传输速率，单位也是“位/秒”（b/s）。带宽是一种理想状态（不受任何干扰，没有任何衰减）下的信道传输速率，而前面介绍的“数据传输速率”是指当前实际的数据传输速率。通常情况下“数据传输速率”永远小于或等于“带宽”值（事实上一般不可能等于带宽的，因为不可能完全不受干扰，线路性能也一般不可能完全达到理想状态）。

可以打这样一个比方，如一条河流它能承受的最大水流速度为 $1000m^3/s$，但现在由于河里的水不多，实际的水流速度仅 $100m^3/s$。或者原来这个河在设计之初是可以承载 $1000m^3/s$ 的水速，但是由于年久河床被淤泥堵塞，当前只能承载 $100m^3/s$ 的水速。这都可与“数据传输速率”和“传输带宽”的关系进行类比的。

4.4.2 数字信号不失真传输的最大传输速率限制

任何一个传输信道所能承受的信号频率（或者说是信道带宽）都是有限的，就像任何一条河流也只能承受一定峰值的水流量一样。在信道中，超出限定频率范围的信号都将被直接滤出，不能通过信道。

在数据通信中有数字信号传输和模拟信号传输之分（对应也就有数字信道和模拟信道之分了），而且数字信号的传输要求与模拟信号的传输要求不同：模拟信号的传输要求接收端无波形失真，而数字信号的传输要求接收端无差错地恢复成原来的二进制码（可以允许接收波形失真，只要不影响正确恢复信号编码即可）。在模拟信号传输中，为了达到不失真的效果，就需要对由模拟信号转换成数字信号过程中的采样频率进行限制；而在数字信号传输中，就需要对信号传输速率进行限制。本节仅介绍数字信号传输中信号传输速率的限制。

在计算数字信道中最大传输速率方面有两个非常有名的准则或定理，那就是“奈奎斯特准则”（简称“奈氏准则”）和“香农公式”。

1. 奈奎斯特准则

由于数字信号（这里特指没有被调制的原始基带数字信号）的频带可以非常宽（从直流一直到无限高的频率），但其主要能量则集中在低频段，而且数据通信中的线缆传输信道是只允许比较低的频率成分通过的理想低通信道，高频成分将被滤去，这就造成了输出波形的失真。失真的输出波形顶部变圆，底部变宽（称为“波形拖尾”），使得一个码元的波形展宽到其他码元位置，影响到其他码元，这种影响称为“码间干扰”。

出现码间干扰后，波形的拖尾可能是正，也可能是负：如果某个码元内所有的拖尾部分相加后是正值，而且达到门限判决电平（也就是判断信号电平为1或为0的基准电平）就可能将“0”误判为“1”码；反之，如果所有的拖尾相加后在某个码元内的值是负的，就可能将“1”码误判为“0”码。

【说明】“理想低通信道”是指只要信号频率不超过某个上限值都能够不失真地通过此信道，而频率超过该上限值的所有高频分量都不能通过该信道。与之相对的还有一种称为“理想带通矩形信道”，它是指仅允许在上下限之间的信号频率成分不失真地通过，其他频率成分不能通过。

要使信道中传输的数字信号不失真，信道中的数据传输速率必须限制在某个范围之内。1924年，美国的著名物理学家——奈奎斯特（Nyquist）经过多次实验得出：为了确保数字信号不失真地传输，在理想低通信道下实际的最高码元传输速率（即无码间干扰的最高波特率，单位为Baud）必须在以下公式计算结果之内：

$$\text{最高码元传输速率（MaxRB）}=2W \qquad ③$$

其中，W是理想低通信道的带宽，单位为赫兹（Hz）；MaxRB为最高码元传输速率，单位为Raud（波特）。这就是著名的奈奎斯特第一准则，指出了在理想低通信道中，为了确保数字信号的不失真传输，信道带宽和最大码元传输速率之间的关系。

从该公式可以得出，每赫兹带宽的理想低通信道（就是基带传输信道）的最高码元传输速率是每秒2个码元。若码元的传输速率超过了这一准则所给出的数值则将出现码间干扰，以致在接收端无法正确判定码元是1还是0。

【说明】奈奎斯特是美国的一位著名物理学家。他1889年出生于瑞典，1907年移民到美国并于1912年进入北达克塔大学学习，1917年在耶鲁大学获得物理学博士学位，1917年至1934年在AT&T公司工作，后转入贝尔电话实验室工作，最终于1976年在美国德克萨斯州逝世。奈奎斯特对信息论做出了重大的贡献。

例如低通信道带宽为2000Hz时，最高码元传输速率就为4000 Baud，即每秒最多可传送4000个二进制码元。反过来，根据最大码元传输速率也可以算出信道带宽。如一路数字电话速率为64Kb/s（因为是数字电话，采用的是二进制，所以比特率与波特率是相等的，所以此电话的波特率为64000 Baud），则根据以上公式可以计算出无码间干扰的信道带宽为32kHz。

另外，在奈奎斯特第一准则中也同时得出，在理想带通矩形信道（也就是频带传输中的信道）下的最高码元传输速率与信道带宽具有相等值关系，即每赫宽带的带通信道的最高码元传输速率为每秒1个码元。

2. 香农公式

以上奈奎斯特第一准则描述了有限信道带宽、无噪声信道的最大数据传输速率与信道带宽的关系。美国的另一位著名物理学家——克劳德·香农（Claude Elwood Shannon）也在数据通信领域有权威的发现，那就是他的“香农公式”。它描述了有限信道带宽、有随机热噪声信道（更接近真实环境）的最大传输速率与信道带宽、信噪比（信号平均功率与噪声功率之比）之间的关系。

香农公式指出：在有随机热噪声的信道上传输数据信号时，最大数据传输速率Rmax与信道带

宽 B、信噪比 S/N 的关系如以下公式所示：

$$R_{max}=B\times\log_2(1+S/N) \quad ④$$

式中，R_{max} 的单位为 b/s，信道带宽 B 的单位为 Hz，信噪比 S/N 通常用 dB（分贝）表示，可用 $10\times\log_{10}(S/N)$公式来计算信噪比的分贝数。若已知 S/N=1000，则根据公式可以很快得出用分贝数表示的信噪比为 30dB。如果再已知信道带宽 B=3000Hz，则 $R_{max}\approx30$Kb/s。它表示对于带宽只有 3000Hz 的通信信道，信噪比在 30dB 时，无论数据采用二进制还是更多的离散电平值表示，都不能用超过 30Kb/s 的数据传输速率。

4.4.3 模拟信号不失真还原的最小采样频率限制

上节介绍了数字信号不失真传输时信道中的最大传输速率限制，本节要介绍的是模拟信号传输中在接收端不失真还原的最小采样频率限制。

我们知道，在计算机设备中只能接收数字信号，如果在信道中传输的是模拟信号（通过公共交换电话网进行的远程拨号连接时在电话线中传输的都是模拟信号），在接收端必须进行模/数转换，生成数字信号；同样如果将在一个计算机设备中的数字信号还原到某条模拟线路中进行传输，则又要进行数/模转换，还原成原来的模拟信号。这里最关键的是由模拟信号转换成数字信号，因为在这个转换过程中要确保数据不失真。

因为模拟信号是随时间连续变化的，而数字信号是离散的，所以在模/数转换过程中就需要离散地在模拟信号上抽取一定的信号样本，形成数字信号样本，以便可以不失真地代表原始数据本身，也使得抽取的数字信号样本在需要时能还原为原始的模拟信号。这就涉及到一个“采样频率”（简单地说就是每隔多少时间提取一次信号波）的问题。

从信号处理的角度来看，采样定理描述了两个过程：其一是采样，这一过程将连续时间信号转换为离散时间信号；其二是信号的重建，这一过程是将离散信号还原成连续信号。连续信号在时间（或空间）上以某种方式变化着，而采样过程则是在时间（或空间）上以 T 为单位间隔来测量连续信号的值。T 称为采样间隔。

在实际中，如果信号是时间的函数，通常他们的采样间隔都很小，一般在毫秒、微秒的量级。采样过程产生一系列的数字称为样本，代表了原来的信号。每一个样本都对应着测量这一样本的特定时间点，而采样间隔的倒数 $1/T$ 为采样频率（f_s），其单位为样本/秒，即赫兹（Hz）。信号的重建是对样本进行插值的过程，即从离散的样本 $x[n]$中用数学的方法确定连续信号 $x(t)$。要确保信号不失真地恢复，最低的采样频率需要达到多少呢？采样过程所应遵循的规律又称取样定理、抽样定理。采样定理说明采样频率与信号频谱之间的关系，是连续信号离散化的基本依据。

采样定理是 1928 年由美国物理学家奈奎斯特首先提出来的，因此称为“奈奎斯特采样定理”。1933 年由苏联工程师科捷利尼科夫首次用公式严格地表述这一定理，因此在苏联文献中称为“科捷利尼科夫采样定理”。1948 年信息论的创始人香农对这一定理加以明确地说明并正式作为定理引用，因此在许多文献中又称为“香农采样定理”。

上述采样定理又有许多表述形式，但最基本的表述方式是“时域采样定理”和“频域采样定理”。

在计算机网络的数据通信中，主要用到的是“时域采样定理”。

“时域采样定理”中规定，频带为 F 的连续信号 $f(t)$可用一系列离散的采样值 $f(t_1)$,$f(t_1\pm\Delta t)$,$f(t_1\pm2\Delta t)$,...来表示，只要这些采样点的时间间隔 $\Delta t\leqslant 1/2F$ 便可根据各采样值完全恢复原来的信号 $f(t)$。简单地讲就是，在进行模拟/数字信号的转换过程中，当采样频率 f_{smax} 大于或等于信号中最高频率 f_{max} 的 2 倍时（$f_{smax}\geqslant 2f_{max}$），采样之后的数字信号完整地保留了原始信号中的信息，原来的连续信号才可以从采样样本中完全重建出来，否则采样信号后的频率将会混叠。一般实际应用中保证采样频率为信号最高频率的 5～10 倍。

从以上定理可以得知，如果已知信号的最高频率 f_H，就可以得出保证完全重建信号的最低采样频率是 $2f_H$。相反，如果已知采样频率，则可以得出保证信号完全重建所允许的最高信号频率。但需要注意的是，这里有一个条件，那就是被采样的信号必须是有限带宽的，信号中高于某一给定频率的成分必须是 0 或者至少非常接近 0。只有这样，在重建信号时这些频率成分的影响才可以忽略不计。如果被采样的信号不是有限带宽的，采样后信号的频率就会产生重叠，即高于采样频率一半的成分将被重建成低于采样频率一半的信号，最终将导致信号不能被完全恢复。

“频域采样定理”中规定了对于时间上受限制的连续信号 $f(t)$的采样频率。因为比较复杂，而且在数据通信中比较少用，故在此不作介绍。

4.5 矩形脉冲数字信号基本波形

在本章前面我们提到了，数字基带传输模式中传输的信号是要经过编码的数字信号。数字基带信号的类型有很多，常见的有矩形脉冲、三角波、高斯脉冲和余弦脉冲等。其中最常用的是矩形脉冲，因为矩形脉冲易于形成和变换。本节先介绍矩形脉冲数字信号中几种常见的波形和编码方式。

在矩形脉冲数字信号中主要有以下 4 种波形：单极性不归零波形、单极性归零波形、双极性不归零波形、双极性归零波形。

4.5.1 单极性波形和双极性波形

在以上 4 种矩形脉冲波形中首先要理解两个概念，那就是什么是“单极性”和“双极性”。

1. 单极性

“单极性”就是波形中仅用正（或负）电平值来表示信号中的二进制“1”（通常是以正电平表示），0 电平表示信号中的二进制“0”。这里所说的“电平”也就是通常所说的电流，这样一来，在单极性中凡是有电流的脉冲部分就代表传输的是二进制数码“1”，无电流部分代表传输的是二进制数码“0”，很容易区分。

【说明】在数据传输中通常要求不要有 0 电平，因为 0 电平与没信号的表现是一样的，这时就不知道是网络故障造成的，还是真的没有电平，传输的是数码 0。

在“单极性”波形中判断对应电平信号是 1 还是 0（也就是通常所说的“判决电平”）的依据就是信号电平矩形脉冲幅度（即信号波的“振幅”）的 1/2 处的信号电平值，大于或等于振幅的 1/2

时就用 1 表示，小于振幅的 1/2 时就用 0 表示。

之所以称之为“单极性”，就是因为在单极性脉冲信号中仅使用了其中一个极性（要么用正，要么用负，一般用正极性）来代表所传输的数码“1”。如图 4-12 上图所示是一个信号代码为 0101101010 的单极性矩形脉冲波示例。

2. 双极性

“双极性”是指分别用正和负电平值表示信号中的二进制“1”和“0”，而且正、负脉冲幅度一样。此时无论传输的是数码 1 还是数码 0 都有脉冲电流。双极性脉冲信号中的判决电平就是零电平，信号电平值大于 0 时用 1 表示，信号电平值小于 0 时用 0 表示。

之所以称之为“双极性”就是因为在双极性脉冲信号中同时使用了正、负两种极性来表示所传输的二进制数码值，即当电平值为正时代表传输的是 1，为负时代表传输的是 0。如图 4-12 下图所示是一个信号代码为 0101101010 的双极性矩形脉冲波示例。

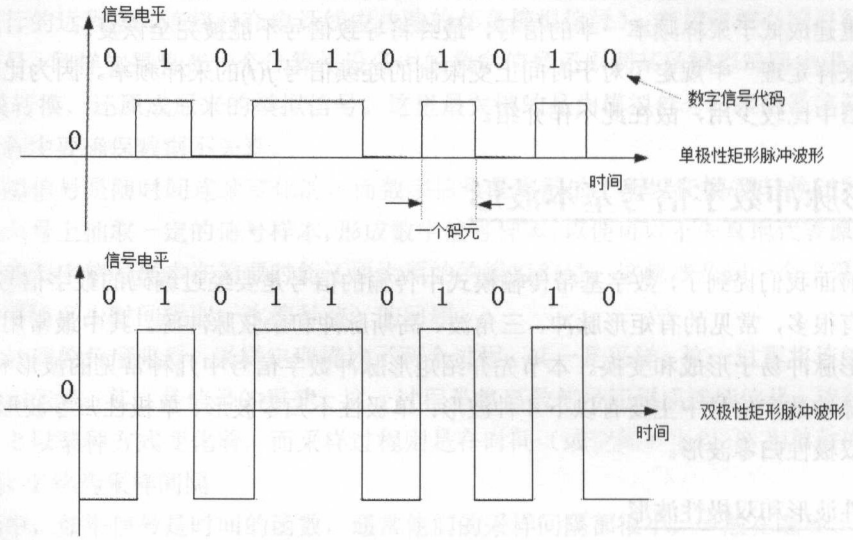


图 4-12　单极性矩形脉冲波形和双极性矩形脉冲波形

显然，在双极性波形中是没有零电平的，数码 0 都是有电平的，所以更适合于数据传输。

4.5.2　归零码和非归零码

上节介绍的两种波形中，单极性波形是依据在一个码元的时间内发出或不发出电流来判定所传输的是数码 1 还是 0，而双极性波形是依据在一个码元的时间内发出的是正电流还是负电流来判定所传输的是数码 1 还是 0。但它们每个所传输的数码都占用了整个码元的宽度，所以这两种编码都属于“全宽码”。

这种全宽码波形存在一个码元难以识别的问题，因为当连续发送 1 码或者 0 码时，就会使某一位码元与其下一位码元之间没有间隙（因为持续有电流），容易使接收方误认为对应的信号电平是

稳定的直流电平。正因为如此，在数据通信中一般不是直接采用这种原始波形进行传输，而是采用了“归零码”这种编码方式。这种不重新编码的波形称为“非归零码”，在传输中难以确定一位的结束和另一位的开始，需要用某种方法使发送器和接收器之间进行定时或同步。

理解什么是“归零码”和“非归零码”的关键就是要理解什么是“归零”。“归零”是指信号电平在一个码元宽度内（通常是在 1/2 个码元时）信号脉冲电平必须回归为零电平（也就是无电流），直到该码元宽度结束。“非归零码”就是不对脉冲信号波进行任何编码，原来是什么样子，传输的就是什么样子，整个码元宽度的电平都是一样的，不强制归零。

归零码的典型代表就是本章后面将要介绍的“曼彻斯特码”和“差分曼彻斯特码”，非归零码的典型代表就是本章后面将要介绍的“AMI 码”“HDB3 码”和“CMI 码”。

归零码的优点是：容易区分码元之间的界限，而且可以通过 1/2 码的归零提取位同步信号，容易实现位同步；缺点是：信息码有效率低（每码元只有 1/2 为有效信息）。非归零码的优点是：不改变原始信号；缺点是：不容易区分码元之间的界限，但可以通过其他方式来解决。

4.5.3 单极性归零码和双极性归零码

因为矩形脉冲波有单极性和双极性之分，归零码对应有 “单极性归零码”和“双极性归零码”，非归零码对应有“单极性非归零码”和“双极性非归零码”。一般来说，直接采用“单极性非归零码”和“双极性非归零码”都比较少。“单极性非归零码”主要适用于极短距离传输，更多是需要再进行其他形式的编码，如下节将要介绍的各种传输码都是在“单极性非归零码”的基础上改进的，最终都变成了双极性的编码。“双极性非归零码”常在 CCITT 的 V 系列接口标准或 RS-232 接口标准中使用。

1. 单极性归零码

从以上对“归零”这两个字的理解我们可以很快理解“单极性归零码”的含义了。“单极性非归零码”仅采用一个极性，并且“1”码表示有信号（正电平或负电平，这取决于所选择的极性），“0”码表示无信号（零电平）。

把“单极性非归零码”转换成“单极性归零码”就很简单了，只需要把“1”码的脉冲电平在其码元宽度内（通常是在 1/2 个码元时）从正电平或负电平回归到零电平，一直持续到该码元结束；原来为“0”码的保持零电平不变，因为它已是零电平了，再回归零电平还是零电平。如图 4-13 所示是一个“单极性非归零码”转换为“单极性归零码”的示例。

从以上可以看出，“单极性归零码”的主要优点是可以直接提取同步信号，因为在出现连续的 1 码元时都会有一个小段（通常为 1/2 个码元宽度）的零电平间隙，不会出现任何两个码元连续的正电平（或负电平），比较容易区分有信号时两个码元之间的界限。但它仍有一个不足，就是当下一个码元也是无信号的零电平时，则两个码元仍会是连续的零电平，仍比较难实现数据传输的同步，因为仍不能区分两个连续 0 码元的界限。而且单极性码中采用了零电平，包括了直流和低频成分，不利于在信道中传输，所以通常是采用下面的双极性归零码波形。

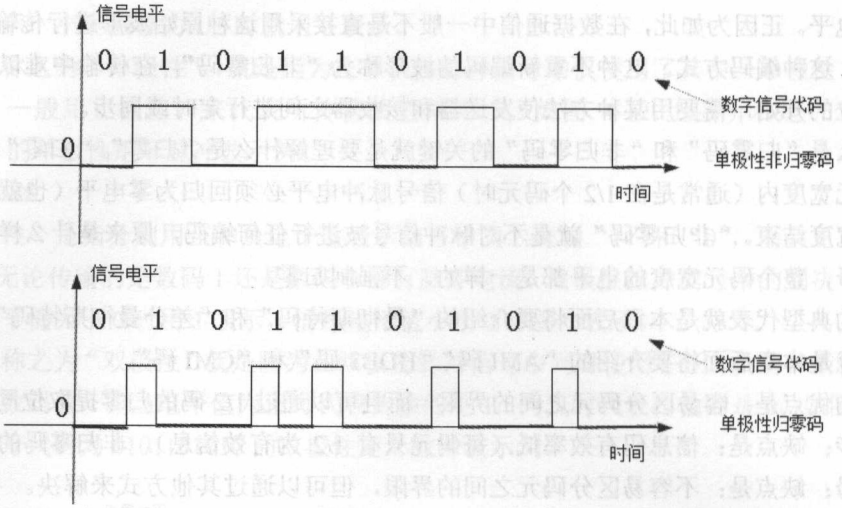


图 4-13 “单极性非归零码”转换为“单极性归零码”的示例

2. 双极性归零码

由“双极性非归零码”转换成“双极性归零码”的原则也是按照 4.5.1 节中介绍的对“归零”两个字的理解，只不过因为“双极性非归零码”中的信号不采用零电平，而是采用正电平来表示 1 码元，负电平表示 0 码元，所以此时“双极性归零码”中的两个极性（不管是正电平还是负电平）的码元电平都必须在一个码元宽度内（通常也是在 1/2 个码元时）回归零电平，一直持续到该码元结束。如图 4-14 所示是一个“双极性非归零码”转换为“双极性归零码”的示例。

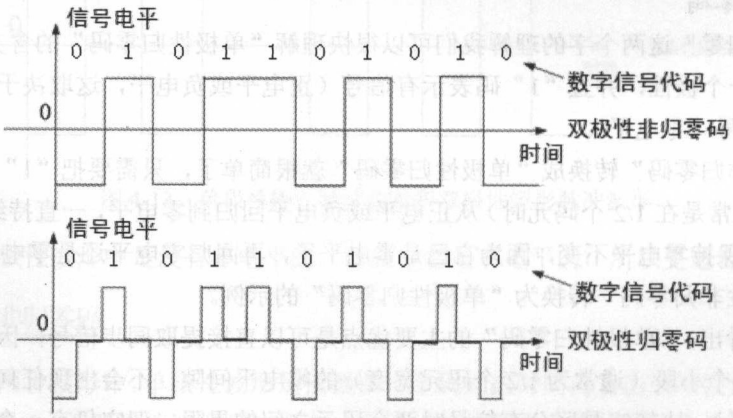


图 4-14 “双极性非归零码”转换为“双极性归零码”的示例

从以上可以看出，在“双极性归零码”中永远不会出现任何两个码元电平的连续，不管是正电平还是负电平，在其码元宽度（通常为 1/2 个码元宽度）间都会回归到零电平，每个码元之间都有

一小段（通常为 1/2 个码元宽度）零电平的间隙，非常容易区分不同码元，也非常容易实现数据的同步传输。另外，尽管双极性码编码前不采用零电平，仅有正、负电平，但在编码后仍含有较多的零电平直流成分，所以也不宜在信道中传输。

如果把图 4-12 至图 4-14 合并在一起，就更容易通过对比看出它们之间的区别，以及非归零码转换成归零码的规则了，如图 4-15 所示（它们传输的信号代码相同）。

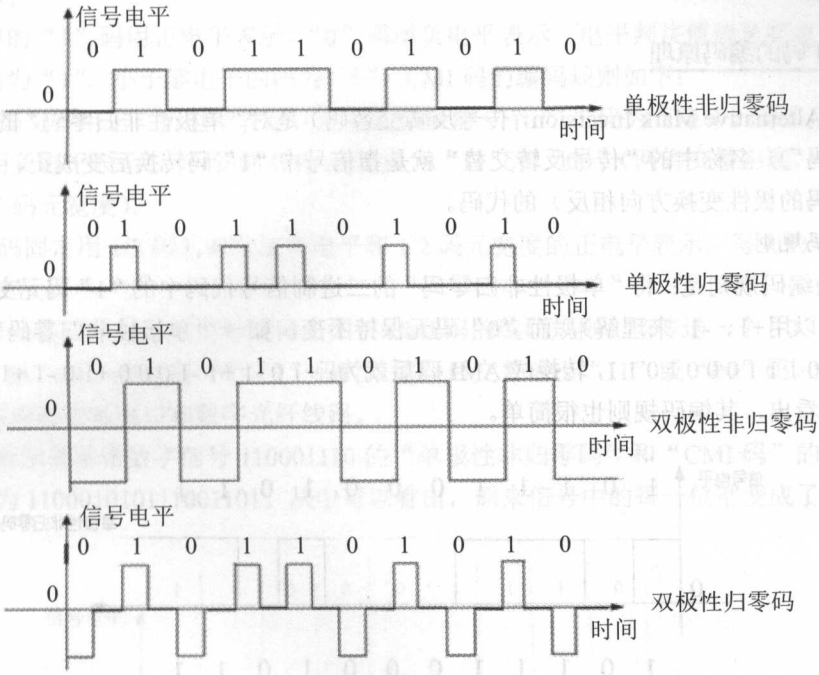


图 4-15 单/双极性非归零码和单/双极性归零码的比较

4.6 数字基带信号传输码

在实际的基带传输系统中，并不是所有编码类型的电信号波形都能在信道中传输。例如在上节中介绍的含有直流分量和较丰富低频分量的单极性基带波形就不适宜在低频传输特性差的信道中传输，因为其功率较弱，在传输过程中有可能造成信号严重畸变。另外，当信息代码中包含有长串的连续“1”或“0”时，非归零波形呈现出连续的固定电平，会出现无法获取同步的问题。单极性归零码在传送连“0”信息代码时也会存在同样的问题。因此，信息码在进行传输之前必须经过码型变换，变换为适用于信道传输的码型，这就是通常所说的“传输码”，也称“线路码”或“信道编码”。

传输码型的选择：要能从基带信号中提取定时信息（位同步脉冲信息），所以要求传输码型应含

有（或者经变换后含有）时钟频率分量（不能出现零电平），而且编码转换后不能出现过多的连续“0”码或“1”码，否则提取的时钟信号就会很不稳定，引起同步偏移；要无直流分量，低频成分少，否则影响信号在信道中的传输；信号中高频分量尽量少，以节省传输频带并减少码间串扰。

常见的传输码型有：传号反转交替码（AMI 码）、三阶高密度双极性码（HDB3 码）、传号反转码（CMI 码）、数字双相码（曼彻斯特码）和差分曼彻斯特码。上节介绍的两种“归零码”也是传输码类型。

4.6.1 AMI 码的编码原理

AMI（Alternative Mark Inversion，传号反转交替码）是对“单极性非归零码”的变形（也称“非归零 AMI 码”），名称中的“传号反转交替”就是指信号中“1”码转换后变成正、负极性交替（但相邻“1”码的极性变换方向相反）的代码。

1. 编码规则

AMI 的编码规则是：将“单极性非归零码”的二进制信号代码中的“1”码元交替地替换为正、负电平（可以用+1、-1 来理解），而“0”码元保持不变。如一“单极性非归零码”中的二进制信号代码为 1 0 1 1 1 0 0 0 1 0 1 1，转换成 AMI 码后就为：+1 0 -1 +1 -1 0 0 0 +1 0 -1 +1，波形如图 4-16 所示。可以看出，其编码规则也很简单。

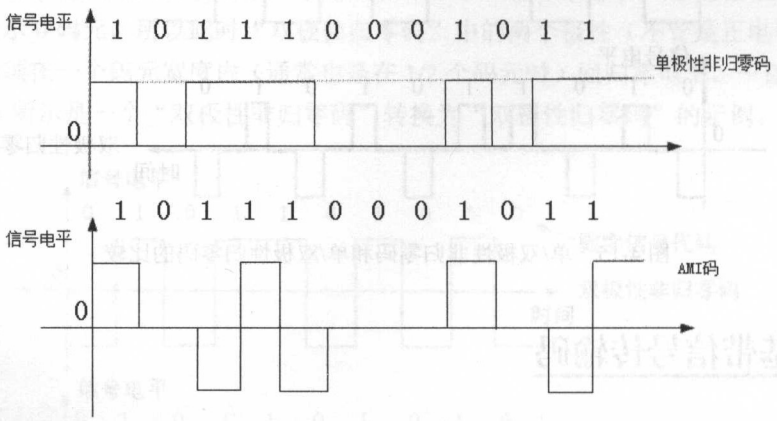


图 4-16 AMI 码与单极性非归零码的比较

2. 主要优缺点

AMI 码的优点是：由于“1”码元采用了正、负电平交替，所以 AMI 码的有信号部分的频谱中不含直流成分，高频和低频成分也很少。另外，AMI 码的编译码电路简单，便于利用传号极性交替规律观察误码情况。鉴于这些优点，AMI 码是 CCITT 建议采用的传输码型之一。

AMI 码的主要缺点是：当原二进制信号代码出现连续的“0”码时，信号电平会长时间不变，造成同步困难，不能纠错。解决这个问题的有效方法之一是采用后面将要介绍的 HDB3 码，它是 AMI 码的改进码型。

4.6.2 CMI 码的编码原理

CMI（Coded Mark Inversion，传号反转码）是一种双极性二电平非归零码，也称 1B2B 码（即一位信息码用二位表示）。所谓“二电平”就是一个数码用两个电平来表示，即我们经常在网上见到的 1B2B 类型编码。

1. 编码规则

双极性码中的“1”码用正电平表示，“0”码用负电平表示，电平判决值就是零电平。也就是大于零电平的码为“1”，小于零电平的码为“0”。CMI 码的编码规则如下：

- “1”码交替用正、负电平表示（也就是如果前一个“1”码为正电平，那么后一个“1”码则用负电平表示），得到的 CMI 码对应为“11”和“00”（两者交替出现，两个电平各占半个码元宽度）。
- “0”码固定用 1/2 码元宽度的负电平和 1/2 码元宽度的正电平表示，得到的 CMI 码恒定为“01”。

总结以上两条编码规则可以简单地得出如下 CMI 码的编码规则：当传送一个“0”码时，编码后输出固定的“01”码；当传送一个“1”码时，编码后交替输出“11”或“00”码。CMI 码的典型应用是调制解调器设备接口和数字光纤线路。

如图 4-17 所示是基带数字信号 110001110 的“单极性非归零码”和“CMI 码”的比较。最终得到的 CMI 码为 110001010111001101。从中可以看出，原来信号中的每一位都变成了两位。

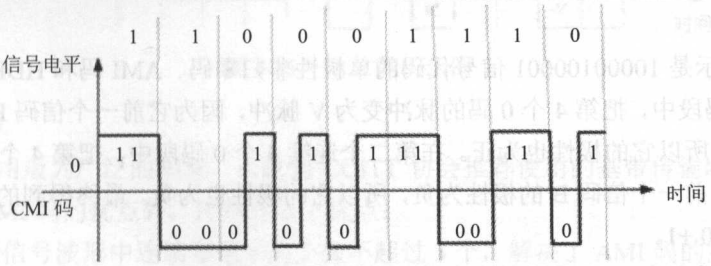


图 4-17 CMI 码转换示例

2. 主要优缺点

CMI 码的主要优点包括：

- 具有较少直流分量（没有零电平，只有正、负电平），而且有频繁出现的波形跳变，便于恢复定时信号（即便于同步）。
- 具有一定的检错能力（但不具有纠错能力），因为“1”码交替地用“11”和“00”来表示，而“0”又固定用“01”表示。这样一来，正常情况下，转换后的编码不可能出现连续的 10、00 和 11，这样一来就可以用来检测因信道传输而产生的错误。
- 编译码电路简单，容易实现。

CMI 码的主要缺点是：需要占用双倍的带宽，因为一位码元转换成两位了。

4.6.3 HDB3 码的编码原理

为了利用 AMI 码的优点并克服其缺点，有科学家就提出了许多种类的改进 AMI 码（所以 HDB3 码也是对“单极性非归零码”的变形码）。其中的典型代表就是各种高密度双极性码 HDBn，HDB3（High Density Bipolar of order 3 code，三阶高密度双极性码）码就是高密度双极性码集中最典型的一种，最关键的特性就是可以确保信号波形中连续零电平（其实也就是码元连续为 0 的个数）的码元数不超过 3 个。

1. 编码规则

HDB3 码的编码规则比较复杂，具体如下：

（1）先把基带数字信号代码变换成 AMI 码，然后检查 AMI 码中的连“0”码元的情况，如果信号码中的连续“0”码的个数不超过 3，则这时的 HDB 3 码就是原来的 AMI 码。如一个二进制信号代码为 1 0 0 1 1 0 1 1 0，转换成的 AMI 码为+1 0 0 -1 +1 0 -1 +1 0，从中可以看出连续“0”码最多只有 2 个，没有超过 3 个，所以该信号的最终 HDB3 码与其 AMI 码一样。

（2）当信号 AMI 码中出现连续 4 个或 4 个以上“0”码元时，则将每 4 个（就是每 4 个一组）连续“0”码元的第 4 个“0”转换成非 0 脉冲，记为+V 或-V（是取“破坏”之意的 Violation 单词的第一个字母），称为“破坏脉冲”。原来的信号码元序列中所有的“1”码称为“信码”，用符号 B 表示。至于第 4 个“0”码是转换成+V 码还是-V 码，这里有两个原则必须同时满足：①V 码必须与前一个信码 B 极性相同（同时为正或同时为负）；②相邻 V 码（包括+V 码和-V 码）的极性必须相反。

如图 4-18 所示是 10000100001 信号代码的单极性非归零码、AMI 码和 HDB3 码的比较。在第一个连续 4 个 0 码段中，把第 4 个 0 码的脉冲变为 V 脉冲，因为它前一个信码 B（也是前一个“1”码）的极性为正，所以它的极性也为正。在第二个连续 4 个 0 码段中，把第 4 个 0 码的脉冲也变为 V 脉冲，但因为它前一个信码 B 的极性为负，所以它的极性也为负。最终得到的 HDB3 码就为+1 0 0 0 +V -1 0 0 0 -V 0 +1。

（3）如果通过变换 V 码后得到的波形还不能同时满足以上两个原则，则需要将对应的 4 个连续“0”码的第一个“0”码转换成与该段 4 个连续“0”码的 V 码同极性的补信码，用符号 B’表示。最终使得 V 码与前一个信码（包括 B 码和 B’码）的极性相同，相邻 V 码的极性以及相邻信码的极性都是相反的。

如现在信号代码为 100001100001，如果仅把第 4 个“0”码转换成对应级性的 V 码，则得到如图 4-19 所示第 3 个波形的不正确 HDB3 码。因为此时第二个 V 码的极性为负，不与它前面一个 B 码极性相同，不满足第 2 步中说到的第①个原则。此时就要将第二个连续 4 个 0 码的码段第一个“0”位置转换为与前一个 B 码极性相反的 B’码。如图 4-19 所示为第四个波形的正确 HDB3 码。这时就全部符合以上规则了。最终的 HDB3 码为：+ 1 0 0 0 +V - 1 +1 -B’0 0 -V -1。

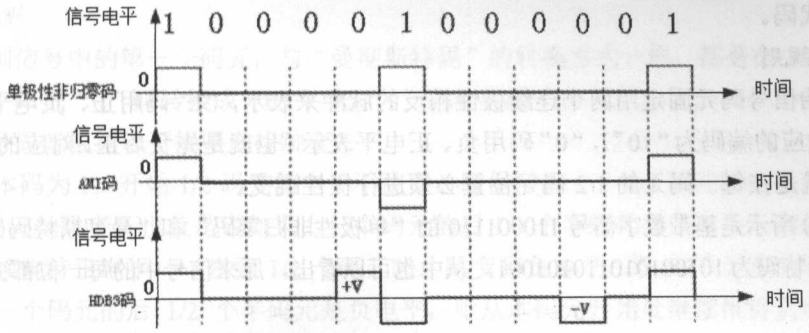


图 4-18 单极性非归零码、AMI 码和 HDB3 码的比较示例

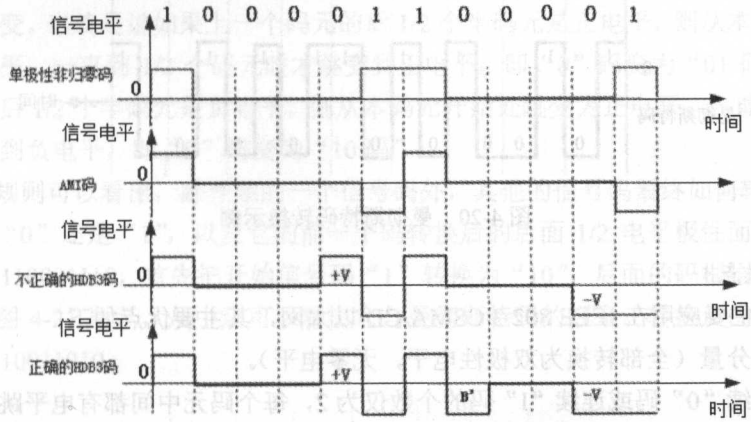


图 4-19 加 B'码的示例

2. 主要优缺点

HDB3 码是应用最为广泛的码型，已成为 CCITT 协会推荐使用的基带传输码型之一。它在保持了前面介绍的 AMI 码的优点外，还具有以下优点：

- 可以确保信号波形中连续零电平的个数不超过 3 个，解决了 AMI 码的主要缺点。
- 消除了 NRZ 码的直流成分，具有时钟恢复和更好的抗干扰性能，这使它更适合于长距离信道传输。
- 具有较强的检错能力。若传输过程中出现单个误码，其极性交替变化规律将受到破坏，因而在接收端根据 HDB3 码这一独特规律特性可检出错误并纠正错误。

HDB3 码的主要缺点是：编码规则比较复杂。

4.6.4 曼彻斯特码的编码原理

“曼彻斯特码”（Manchester Encoding）与上面介绍的 CMI 码类似，它也是一种双极性二电平（1B2B）非归零码。但它是用电压的变化来分辨信号代码中的 0 和 1，而不是直接根据信号电平

来判别信号代码。

1. 编码规则

每个原始信号码元固定用两个连续极性相反的脉冲来表示，“1”码用正、负电平表示（也就是先正后负，对应的编码为“10”），“0”码用负、正电平表示（也就是先负后正，对应的编码为“01”）。其关键特点就是在每一码元的 1/2 码元位置必须进行极性跳变。

如图 4-20 所示是基带数字信号 110001110 的“单极性非归零码”和“曼彻斯特码”的比较。最终得到的曼彻斯特码为 101001010110101001。从中也可以看出，原来信号中的每一位都变成了两位。

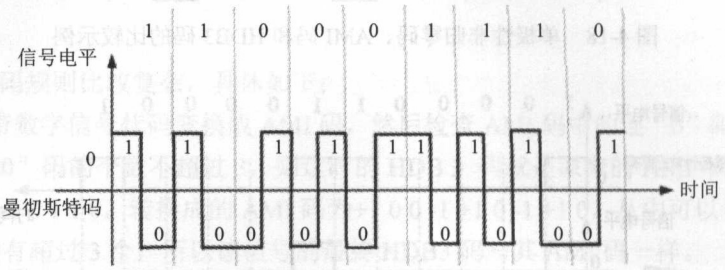


图 4-20　曼彻斯特码转换示例

2. 主要优缺点

曼彻斯特码主要应用在 IEEE 802.3 CSMA/CD 以太网，其主要优点如下：

- 无直流分量（全部转换为双极性电平，无零电平）。
- 最长连续“0”码或连续“1”码的个数仅为 2，每个码元中间都有电平跳变，非常容易实现同步，编译码电路也可以很简单。因为如果原来基带信号是连续的“1”码，转换后都为“10”两个码了，不会出现连续的“1”码，如果原来基带信号是连续的“0”码，转换后都为“01”两个码了，不会出现连续的“1”码或“0”码。仅当原来基带信号是连续的“01”码时才可能出现转换后的连续“0”码；仅当原来基带信号是连续的“10”码时才可能出现转换后的连续“1”码，而且最多仅 2 个连续的“0”码或“1”码。

曼彻斯特码的主要缺点如下：

- 需要占用双倍的带宽，因为它是 1B2B 类型码，每个数码最终要变成两个数码的传输。
- 当极性反转时会引起译码错误，因为不能通过在信码开始处是否有跳变来确定信码是 0 还是 1。

4.6.5 差分曼彻斯特码的编码原理

“曼彻斯特码”当极性反转时会引起译码错误，为解决此问题，又引入了它的衍生码型——差分曼彻斯特码（Differential Manchester Encoding）。“差分曼彻斯特码”是“曼彻斯特码”的一种修改形式，也是每一码元的 1/2 码元位置必须进行极性跳变，其不同之处在于不同位置的码元转换方式有些不一样。

1. 编码规则

对于二进制信号中的第一个码元，与“曼彻斯特码”的转换方式一样，都是在 1/2 码元时进行电平极性跳变，即把“0”码转换成“01”，把“1”码转换成“10”。

后面的信号码就根据以下规则进行跳变：

- 如果本码为 1，开始 1/2 码元的电平不跳变，即保持上一个 1/2 码元的电平不变，然后在 1/2 码元处跳变。也就是说如果上一个码元的后 1/2 个半码元是正电平，则从本码元开始处继续维持正电平，一直到 1/2 个码元时才跳变到负电平，即“1”码变为“10 码”；如果上一个码元的后 1/2 个半码元是负电平，则从本码元开始处继续维持负电平，一直到 1/2 个码元时才跳变到正电平，即“1”码变为“01 码”。
- 如果本码为 0，开始处的电平必须跳变，即与上一个 1/2 码元的电平相反，然后在 1/2 码元处跳变。也就是说如果上一个码元的后 1/2 个半码元是正电平，则从本码元开始处跳变为负电平，一直到 1/2 个码元时才跳变到正电平，即“0”码变为“01 码”；如果上一个码元的后 1/2 个半码元是负电平，则从本码元开始处跳变为正电平，一直到 1/2 个码元时才跳变到负电平，即“0”码变为“10 码”。

从以上转换规则可以看出，除开始的一个信号码外，其他的信号码最终如何转换是不固定的，要根据对应码是“0”还是“1”，以及它的前一个码转换后的后面 1/2 电平极性而定。如原来基带数字信号代码为 110001110，首先把开始信号码“1”转换为“10”，后面的码根据以上规则进行转换，最终结果如图 4-21 所示，从中还可以看出它与曼彻斯特码的区别。转换后的差分曼彻斯特码就为 100101010110011010。

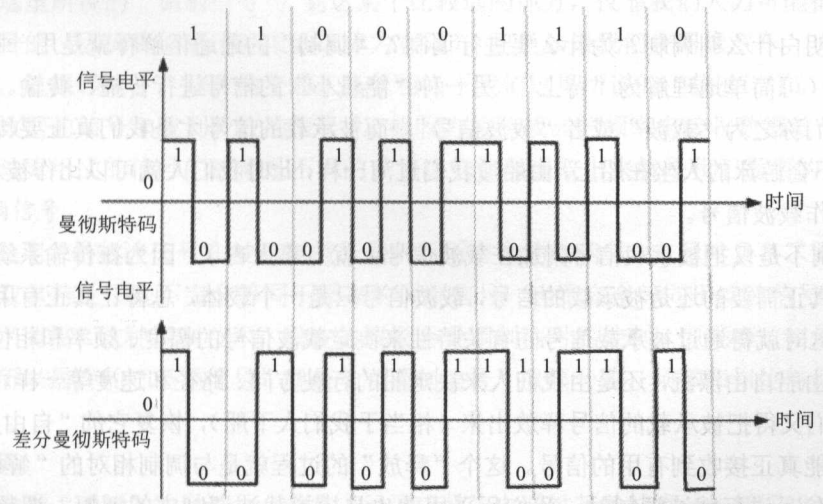


图 4-21 差分曼彻斯特码转换示例

2. 主要优点

差分曼彻斯特码主要用于 IEEE 802.4 令牌总线网络和 IEEE 802.5 令牌环网络。在差分曼彻斯

特码中，每一码元中间的跳变也只用来作为同步的时钟信号，所以也是一种自同步编码。

差分曼彻斯特码的主要优点如下：

- 时钟和数据是分离的，便于数据提取。
- 通过在每个码元位的开始有无跳变来区分信号代码（有跳变为 0，无跳变为 1），在译码时不容易出错。

4.7 信号调制与解调

在计算机网络中，大多数要传输的信号都是具有较低频带的信号，称为“基带信号”。如果将基带信号直接传输称为“基带传输”，这在本章前面已有介绍。但是，大多数信道不适宜进行基带信号的传输，因为这样的低频信号在传输中会产生很大的衰减和失真。

在数字基带传输中可以通过数字编码后在信道中直接传输数字信号，但这也仅适用于近距离传输（如在同一局域网中），在复杂网络环境（如存在不同传输介质和信道类型的广域网）中同样不能直接这样传输。因此需要将基带信号进行调制，变换为适合信道传输的形式。调制是让基带信号去控制载波的某个（或某些）参数，使该参数按照信号的规律进行变化；“解调”就是“调制”的逆过程，是从调制后的信号中恢复原来的调制信号的过程。

4.7.1 调制与解调的关键术语

在“调制”和“解调”过程中涉及到几个非常关键的术语，下面先来理解一下。

1. 调制

我们先要明白什么叫调制？为什么要进行调制？“调制”的最通俗解释就是用一种“能量大”的信号去承载（可简单地理解为“背上”）另一种“能量小”的信号进行传播、传输。承载另一种信号的信号我们称之为“载波”或者“载波信号”，而被承载的信号才是我们真正要传播或者传输的信号。就像不会游泳的人坐在船上，由船渡我们过河一样，此时我们人就可以比作被承载的信号，而船就可以比作载波信号。

但是，调制不是仅把被承载信号附加在载波信号上就万事大吉了，因为在传输系统中要真正发挥作用，用户真正需要的还是被承载的信号，载波信号只是一个载体，总得让真正有用的被承载信号发挥作用。这时就得通过被承载信号的有关特性来决定载波信号的幅度、频率和相位。就像我们坐在船上不是由船自由漂流，还是由我们人来决定船的行驶方向、路径和速度等一样。当信号到了接收端时，我们又得把被承载的信号释放出来（相当于我们人下船），恢复它的“自由身”，只有这样，接收端才能真正接收到有用的信号。这个“释放”的过程就是与调制相对的“解调”。

在对数字信号进行载波调制时，我们所采用的也是模拟载波调制中的调幅、调频和调相这 3 种调制技术，但它们对应已有了另外的名字：ASK（Amplitude Shift Keying，幅度键控）、FSK（Frequency Shift Keying，频率键控）和 PSK（Phase Shift Keying，相位键控）。这里之所以都称之为“键控”，是指在这些调制技术中都是用电键（类似“开关”性质）进行控制的，这是借用了电

报传输中的术语。它们分别对应于利用载波（正弦波）的幅度、频率和相位来承载数字基带信号，可以看做是模拟线性调制和角度调制的特殊情况。

2. 调制信号

“调制信号”就是用于对载波信号进行某个参数特性控制，使其按照自己的对应参数特性变化的原始信号（也是最终要传输的有用信号）。

调制信号通常是低频信号，可以是模拟信号，也可以是数字信号，所以也就有对应的“模拟信号调制”（或“模拟调制”）和“数字信号调制”（或“数字调制”）。“数字调制”与“模拟调制”在本质上没有什么大的区别，都属于正弦波调制，主要是源信号不同，“数字调制”中的源信号为离散型的脉冲数字信号，而“模拟调制”中的源信号为连续型的正弦波信号。在此仅介绍数字信号调制技术。

3. 载波信号

“载波”就是用来载送有用低频“调制信号”的信号波，通常是一种高频信号。通过调制技术就可以把调制信号和载波信号进行叠加，使载波信号的某些参数特性（如信号幅度、信号频率、信号相位等）按调制信号变化。具体是载波的哪个参数会随着调制信号发生变化要视具体的调制类型而定，如果被控制的参数为幅度，则把这种调制称为调幅；如果被控制的参数为频率，则把这种调制称为调频；如果被控制的参数为相位，则把这种调制称为调相。到达接收端后，通过解调技术从已调信号中分离出有用的数据即可。

那么为什么要用载波进行调制呢？因为通常我们要发送的数字信号的频率是比较低的，如果按照本身的频率来传输，信号的衰减比较严重，不利于远距离传输。打个比方，如果我们要送一样物品（相当于这里所说的“调制信号”）到达某个比较远的地方，仅靠我们人力可能很困难，甚至根本无法达到目的（因为我们的速度很低、体力有限），但是如果我们使用像汽车、火车等这类交通工具来运输，就很轻松了，这时的交通工具就相当于我们这里所说的“载波”了。使用载波传输，我们可以将数据的信号加载到载波的信号上，接收方按照载波的频率来接收数据信号，有用义的信号波的波幅与无用义的信号的波幅是不同的，将这些信号提取出来就是我们需要的数据信号。

4. 已调信号

“已调信号”是载波信号在被调制信号调制后所产生的新信号，它同时具备了原来调制信号和载波信号的双重特性（既不完全等同于原来的调制信号，也不完全等同于载波信号）。如在模拟信号调制中的“调幅”这种调制技术，载波的振幅会受调制信号的控制，但频率、相位仍是由载波信号决定的，所以最终的“已调信号”就是与“调制信号”振幅一致，频率和相位与载波信号一致的新信号。

5. 解调

“解调”是“调制”的反过程，是从“已调信号”中通过某种技术（如低通滤波器等）恢复出原来“调制信号”的过程，因为我们最终有用的还是原来的调制信号。之所以要先经过调制过程，目的就是为了便于低频信号在信道中的远距离、高效传输。到了目的地（接收端）后，自然要把它还原出来。

下面各节我们分别对二进制数字信号的几种主要的调制和解调模式工作原理进行介绍。

4.7.2 2ASK 调制原理

ASK（幅度键控）是一种数字幅度调制技术，是指正弦载波的幅度随基带数字信号而变化的数字调制，也称为“通断键控”（on-offkeying，OOK）或者“开关键控”。当数字基带信号为二进制码时称之为二进制振幅键控（2ASK），但它仅适用于单极性数字信号。它是利用代表数字信息“0”或“1”的基带矩形脉冲去键控一个连续的载波，使载波时断时续地输出，因为在单极性波形中数码 0 是无电平输出的。即源数字基带信号为“1”时发送载波信号，源数字基带信号为“0”时发送零电平，也就不发送载波信号。

2ASK 调制就是以不同的波幅来表示源信号中的不同二进位值，相当于用一个如图 4-22 所示的开关电路来控制载波信号的输出。

这个开关就是二进制基带调制信号中的“0”和“1”，当调制信号为“1”码时，相当于开关接通电路，有载波输出，而当调制信号为“0”时，相当于开关断开电路，无载波输出。这是 2ASK 的键控法，另外还有一种模拟法，采用如图 4-23 所示的乘法器把数字基带信号与正弦载波进行乘法运算得出。$\cos\omega_C t$ 为正弦载波信号，$s(t)$为二进制单极性不归零码。

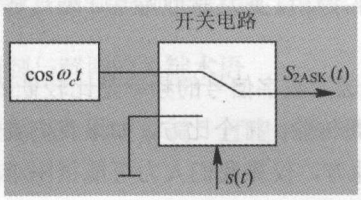


图 4-22　2ASK 键控法调制模型

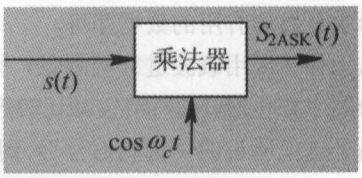


图 4-23　2ASK 模拟法调制模型

如图 4-24 所示是一个 2ASK 调制示例，其中二进制基带调制信号为 1011001，载波为一正弦波。根据以上 2ASK 的调制原理，很快就可以得到最终的已调制信号波形。

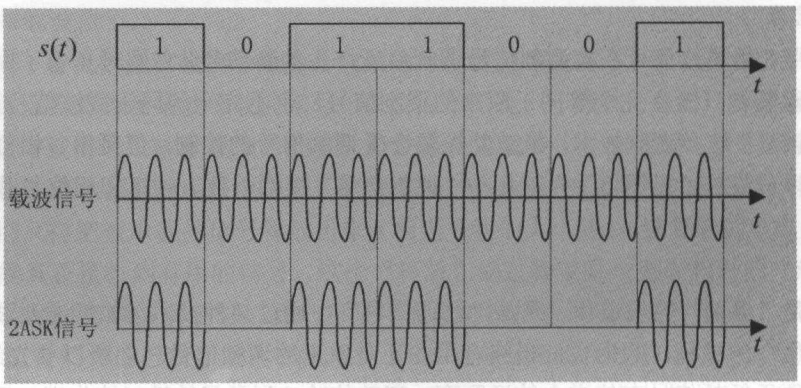


图 4-24　2ASK 调制示例

2ASK 信号是数字调制方式中最早出现的，也是最简单的，但其抗噪声性能较差（因为它仅是根据调制信号来控制输出信号的幅度，并不能过滤干扰信号），因此实际应用并不广泛，但经常作为研究其他数字调制方式的基础。

4.7.3 2ASK 解调原理

与 2ASK 调制有“键控法”和“模拟法”两种方式类似，2ASK 信号解调也有两种方式：“非相干解调”（包络检波法）和“相干解调”（同步检测法）。

相干解调是指要利用乘法器输入一路与载波相干（同频同相）的参考信号，然后再与载波相乘，最终得到解调信号，用于线性调制（指已调信号中的被调参数会随着调制信号成线性变化），如 ASK 和 PSK；非相干解调即不需要提取载波信息（或不需要恢复出相干载波）的一种解调方法，主要用于 FSK，也可用于 ASK。

1. 2ASK 非相干解调

2ASK 非相干解调的基本模型如图 4-25 所示，与模拟信号解调模型的不同点仅仅在于多了一个抽样判决器，因为这是要在已调的连续信号（参见图 4-24 下图所示）中还原为原始的离散二进制数字信号，所以需要对原来的已调信号进行取样。图中的 $y_i(t)$表示的是经过调制后的已调信号，并为各步所生成的信号进行了标注，以便于后面图 4-26 中的波形说明。

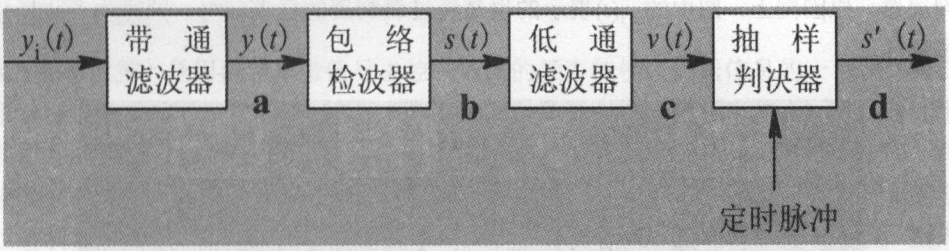


图 4-25 2ASK 非相干解调模型

图 4-26 所示是一个二进制单极性调制信号（源信号）110010001010 调制后经过解调还原出源信号的过程，具体步骤如下：

（1）已调信号 $y_i(t)$经过带通滤波器（主要用于过滤掉已调信号的低频和高频干扰信号）后得到图中的 a 波形。

（2）a 信号再经过包络检波器进行包络检测（“包络”就是外部表现的意思，这里的“包络检测”就是检测出信号的幅度变化的曲线，不管是频率特性的还是相位特性的）后得到了 b 波形。对比 b 和 a 波形可以看出，它是一个极性过滤，因为调制信号是单极性的，所以这里仅允许正极性的波形输出。

（3）b 信号再经过低通滤波器后就得到了 c 波形。这里的低通滤波器用于过滤 b 信号中的高频成分，仅允许符合源信号频率特性的波形输出。

（4）最后经过抽样判决器，对连续型的 c 波形进行取样，还原出原始的离散型调制信号。

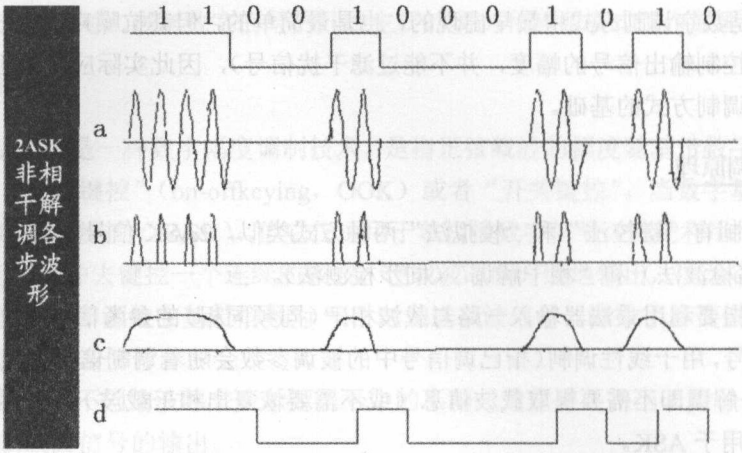


图 4-26　2ASK 非相干解调示例

从以上解调流程可以看出，2ASK 非相干解调没有利用载波信号来影响波形的输出。下面介绍相干 2ASK 解调。

2. 2ASK 相干解调

2ASK 相干解调的基本模型如图 4-27 所示，与模拟信号解调模型的不同点也仅仅在于多了一个抽样判决器，原因同上。图中的 $y_i(t)$表示的也是经过调制后的已调信号，同样为各步所生成的信号进行了标注。

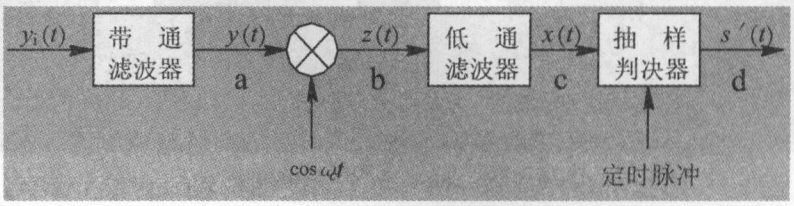


图 4-27　2ASK 相干解调模型

图 4-28 所示也是前面介绍的这个二进制单极性调制信号（源信号）110010001010 已调信号 $y_i(t)$ 经过解调后还原出源信号的过程，具体步骤如下：

（1）已调信号 $y_i(t)$经过带通滤波器（主要用于过滤掉已调信号中的低频和高频干扰信号）后得到图中的 a 波形。这与非相干解调的步骤是一样的。

（2）a 信号再与载波信号 $\cos\omega_c t$ 在乘法器中进行相乘得到了 b 波形。对比 b 和 a 波形可以看出，它不仅是一个极性过滤，而且包含了载波的频率成分，信号更强。

（3）b 信号再经过低通滤波器后就得到了 c 波形。这里的低通滤波器用于过滤 b 信号中的高频成分，仅允许符合源信号频率特性的波形输出。这步与“非相干解调”也一样。

（4）最后经过抽样判决器，对连续型的 c 波形进行取样，还原出原始的离散型调制信号。这步也与“非相干解调”一样。

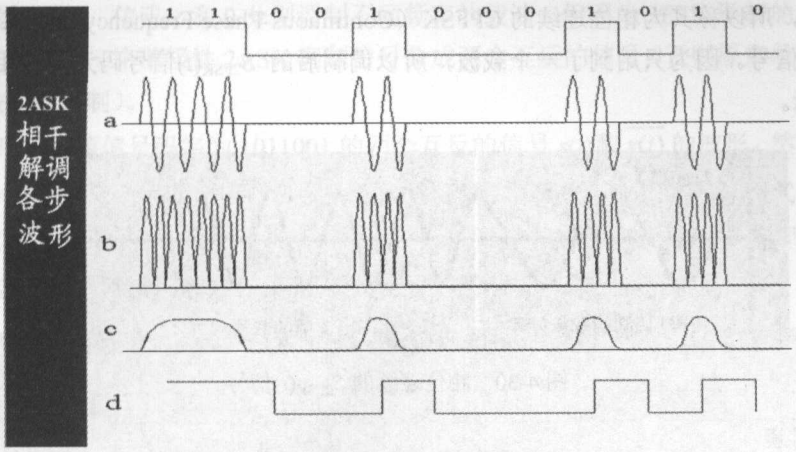


图 4-28 2ASK 相干解调示例

从以上可以看出，“2ASK 非相干解调”和“2ASK 相干解调”在执行步骤上唯一的区别就是第 2 步，“相干解调”借助了载波信号，主要是为了使信号得到加强，更便于在后面的带通滤波器中过滤。

4.7.4 2FSK 调制原理

FSK（频率键控）是用数字基带信号去控制载波的频率特性来传送信息的一种调制方式，但它仅适用于双极性数字信号。它是把信号的振幅、相位作为常量，而把频率作为变量，通过频率的变化来实现信号的识别。如数字信号的“1”码用频率为 f_1 的载波（载波 1）来传送，“0”码用频率为 f_2 的载波（载波 2）来传送，相当于载波在两种不同频率之间进行切换，故又称“频移键控”。

在二进制数字调制中，若正弦载波的频率随二进制基带信号在两个频率点间变化，则产生二进制移频键控信号（2FSK 信号）。2FSK 信号的产生与 2ASK 信号的产生一样，从原理上讲，数字调频既可用模拟调频法来实现，又可用键控调频法来实现。

1. 模拟调频

“模拟调频”又称“直接调频”，是采用模拟调频电路来实现的，利用一个矩形脉冲对载波进行频率调制，使输出得到不同频率的已调信号，基本模型如图 4-29 所示。$S(t)$为调制信号，$S_{2FSK}(t)$为已调的 2FSK 信号。

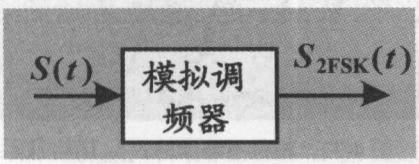


图 4-29 2FSK 模拟法调制模型

使用模拟调频的方法产生的 2FSK 信号对应着两个频率的载波，在码元转换时刻两个载波相位

能够保持连续，所以称其为相位连续的 CPFSK（Continuous-Phase Frequency Shift Keying，连续相位频移键控）信号。因为只用到了一个载波，所以调制后的 $S_{2FSK}(t)$信号码元间的相位是连续的，如图 4-30 所示。

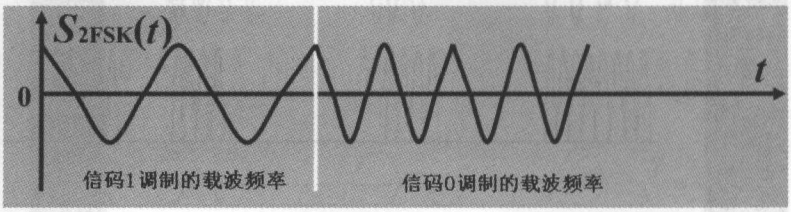


图 4-30　相位连续的 $S_{2FSK}(t)$信号

2. 键控调频

键控调频采用数字键控的方法来实现，是用数字矩形脉冲控制电子开关，使电子开关在两个独立的振荡器（产生两种不同频率的载波 f_1 和 f_2）之间进行转换，从而在输出端得到不同频率的已调信号，基本模型如图 4-31 所示。$S(t)$为调制信号，$S_{2FSK}(t)$为已调的 2FSK 信号。

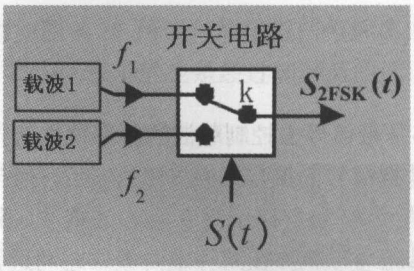


图 4-31　2FSK 键控法调制模型

使用键控调频法时，由于产生 f_1 和 f_2 载频是由两个独立的振荡器实现，所以码元间的相位不一定是连续的，如图 4-32 所示。这种方法的特点是转换速度快、波形好、频率稳定度高、电路不太复杂，在实用中可以用一个频率合成器代替两个独立的振荡器，再经分频链进行不同的分频，也可得到 2FSK 信号。

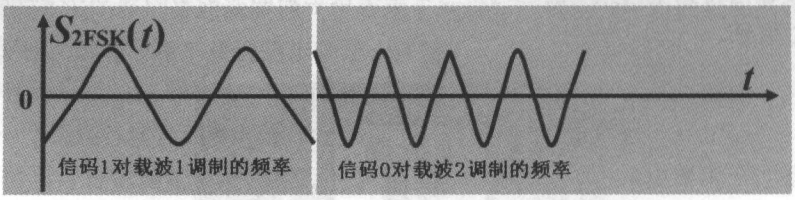


图 4-32　相位不连续的 $S_{2FSK}(t)$信号

3. 2FSK 波形与 2ASK 波形的关系

对比上节介绍的 2ASK 波形和本节中的 2FSK 波形可以看出，2FSK 信号波形可以看做两个

2ASK 信号波形的合成，信码 1 和 0 分别调制不同频率的载波。但因为 2FSK 调制的是双极性数字信号，所以我们可以把一个单极性 2ASK 调制信号分成两个互反的信号（否则原来单极性信号中的 0 码无法对载波进行调制）。

如图 4-33 所示是源信号码字为 1011001 的两个互反的信号 $s(t)$和 $\overline{s(t)}$ 的波形。然后分别用这两路信号作为调制信号，分别对两个不同频率的载波进行调制。其中 $c_1(t)$和 $c_2(t)$分别代表两个不同的载波，$s_1(t)$和 $s_0(t)$是 $s(t)$和 $\overline{s(t)}$ 这两路调制信号分别对 $c_1(t)$和 $c_2(t)$载波进行 2ASK 调制后的波形；再把 $s_1(t)$和 $s_0(t)$这两路信号进行叠加就得到了最终的已调信号 2FSK。

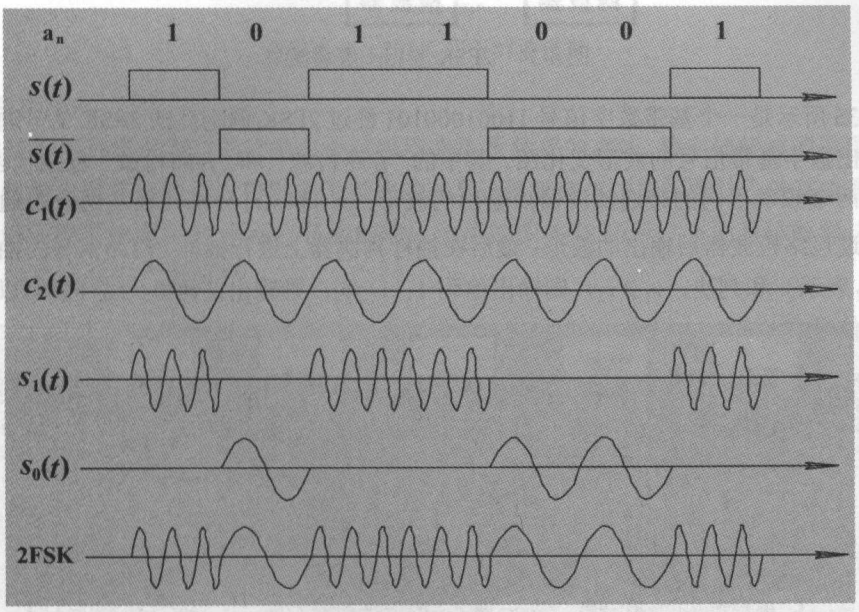


图 4-33　2FSK 调制示例

4.7.5　2FSK 解调原理

2FSK 信号的解调方法很多，如鉴频法、过零检测法、非相干解调法（也称“包络检测法”）和相干解调法（也称“差分检波法”）。这里仅介绍相干解调、非相干解调和过零检测这 3 种 2FSK 解调方法。

1. 2FSK 非相干解调

2FSK 非相干解调（也称包络检测法）模型如图 4-34 所示，其可视为由两路 2ASK 解调电路组成（比较图 4-25）。这里的两个带通滤波器（带宽相同，皆为相应的 2ASK 信号带宽；但中心频率不同，分别为 f_1 和 f_2）起分路作用，用以分开两路 2ASK 信号，经包络检测（这里的“包络检测”就是检测出信号的频率变化的曲线，不管是幅度特性的还是相位特性的）后分别取出互反的 $s(t)$ 和 $\overline{s(t)}$ 两路信号；抽样判决器起比较器的作用，通过把两路包络信号同时送到抽样判决器进行比较就

可以判决输出基带数字信号。如果把上支路 $s(t)$ 和下支路 $\overline{s(t)}$ 的抽样值分别用 v_1、v_2 表示，则抽样判决器的判决准则为：$v_1 \geqslant v_2$，则输出信码 1；$v_1 < v_2$，则输出信码 0。

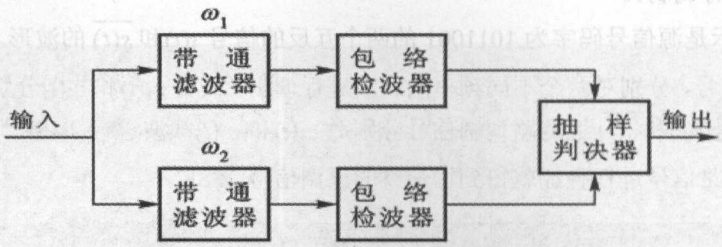


图 4-34　2FSK 非相干解调模型

如图 4-35 所示是一个基带数字信号 11001000101 经过 2FSK 调制后的 2FSK 信号通过非相干解调方法还原出数字基带信号的波形变化图。其中的“f_1 路检波”和“f_2 路检波”分别代表两路 2FSK 信号在经过两个频率不同的带通滤波器后输出的波形；$s(t)$和 $\overline{s(t)}$ 分别代表从两个带通滤波器输出的信号再经过包络检波器后输出的波形；最后在抽样判决器上进行抽样（两路信号的抽样电压值分别用 v_1 和 v_2 表示）和判决，$v_1 \geqslant v_2$，则输出信码 1；$v_1 < v_2$，则输出信码 0。这样就可以还原出原始的数字基带信号。

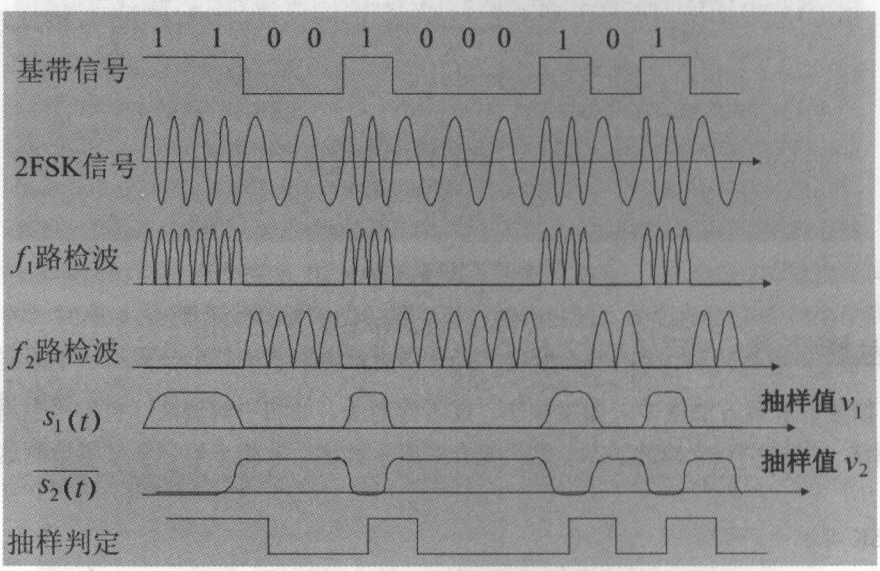


图 4-35　非相干 2FSK 解调示例

2. 2FSK 相干解调

2FSK 相干解调的基本模型如图 4-36 所示。图中的两个带通滤波器是起分路作用。它们的输出分别与相应的同步相干载波相乘，再分别经低通滤波器滤掉二倍频信号，取出含基带数字信息的低

频信号，抽样判决器在抽样脉冲到来时对两个低频信号的抽样值进行比较判决（判决规则与上面介绍的“非相干解调”相同）即可还原出基带数字信号。

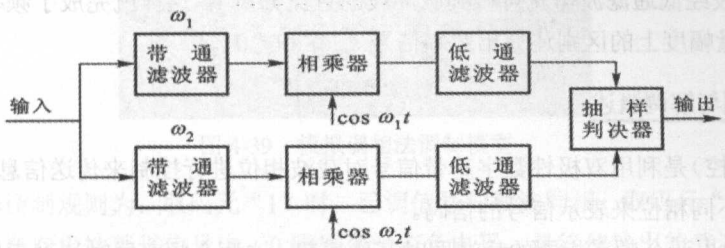


图 4-36 2FSK 相干解调模型

3. 过零检测法

单位时间内信号经过零点的次数多少可以用来衡量频率的高低。数字调频波的过零点数随不同载频而异，故检出过零点数可以得到关于频率的差异，这就是“过零检测法”的基本思想。过零检测法 2FSK 解调模型如图 4-37 所示，各步的输出波形如图 4-38 所示。下面具体解释这些步骤，具体的实现方法在此我们不做研究。

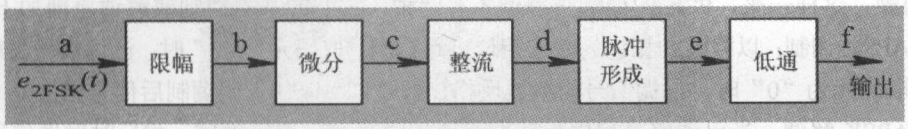


图 4-37 过零检测法 2FSK 解调模型

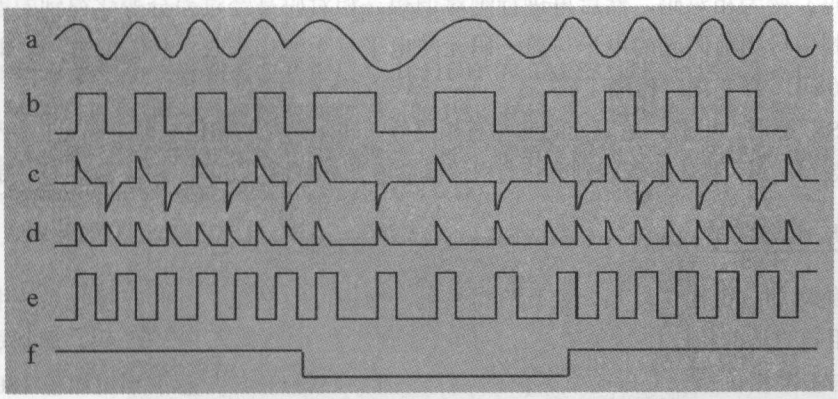


图 4-38 2FSK 过零检测法解调示例

（1）2FSK 输入信号 a 经放大、限幅后产生矩形脉冲序列 b。

（2）b 信号经微分及整流后形成与频率变化相应的尖脉冲序列（分别对应波形 c 和 d），这个序列就代表着调频波的过零点。

（3）d 尖脉冲再触发一脉冲发生器，变换成具有一定宽度的矩形波 e。该矩形波的直流分量便代表着信号的频率，脉冲越密，直流分量越大，反映输入信号的频率越高。

（4）e 矩形波经低通滤波器就可得到脉冲波的直流分量 f。这样就完成了频率—幅度变换，从而再根据直流分量幅度上的区别还原出数字信号“1”和“0”。

4.7.6 2PSK 调制与解调概述

PSK（相位键控）是利用双极性数字基带信号对载波相位进行控制来传送信息的一种调制方式，也就是用载波的不同相位来表示信号的信码。

我们知道，如果两个频率相同的载波同时开始振荡，这两个频率同时达到正最大值，同时达到零值，同时达到负最大值，此时它们就处于“同相”状态；如果一个达到正最大值时，另一个达到负最大值，则称为“反相”。一般把信号振荡一次（一周）作为 360 度。如果一个波比另一个波相差半个周期，则这两个波的相位差 180 度，即反相。因此数字调相就是让载波在两种相位间切换，故又称“相移键控”。

如果调制信号是二进制数字基带信号，则 PSK 称为 2PSK。PSK 分为“绝对相位键控”（Absolute Phase Shift Keying，APSK）和“相对相位键控”（或“差分相位键控”，Differential Phase Keying，DPSK）两种。这样一来，PSK 的调制与解调各有两种，它们的基本调制或解调原理如下：

- APSK 调制：以载波的相位为基准相位进行调制，取码元为“1”时，已调信号与载波同相；取码元为“0”时，已调信号与载波反相，即“1”和“0”码调制后信号相位差 180 度。
- APSK 解调：当已调信号相位与基准相位相同时发送信码“1”，相反时发送信码“0”。
- DPSK 调制：先用差分编码器对数字基带信号进行差分编码，即由绝对码（即原始基带信号码）变为相对码，然后再进行绝对调相。相对码是经过差分变化得到的差分码，是用调制信号当前码元的电平与前一码元的电平有无变化来表示传输的数码。即无变化时传输 0 码，有变化时传输 1 码。
- DPSK 解调：用前、后码元间的相对相位值（即本码元初相位与前一码元初相位之差）来传送数字信息。具体解调规则为：当相位偏移值为 0 时输出 0 码；相位偏移 180 度时输出 1 码。

4.7.7 2APSK 调制原理

2APSK 绝对相位键控是以未调载波的相位为基准相位的 PSK，此时在接收系统中必须有一个与发送系统相同的基准相位作为参考，以识别接收到的是“1”码还是“0”码。调制规则为：当已调载波相位与基准相位相差 0 度时发送信码“1”，当已调载波相位与基准相位相差 180 度时发送信码“0”。2APSK 信号的产生也有模拟调相法和键控调相法两种。

1. 模拟调相法

模拟调相法是把 2APSK 看做双极性不归零码基带信号的数字调幅 2ASK，即通过调制信号与载波的乘积得到已调信号，其调制模型如图 4-39 所示。其中 $s(t)$为数字基带信号，$\cos\omega_c(t)$为载波信号，$S_{2psk}(t)$为已调 2PSK 信号。

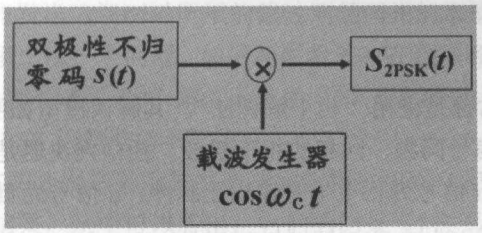


图 4-39 模拟调相法调制模型

模拟调相法的调制规则为：取码元“1”时，已调信号与载波同相；取码元“0”时，已调信号与载波反相。因为在双极性基带信号中，0 码就是对应负电平，最终使输出的载波信号中正变负，负变正。

2. 键控调相法

键控调相法是通过载波在两种不同相位之间进行切换而进行的，其调制模型如图 4-40 所示。其调制规则为：取“1”码输出正向载波，取“0”码输出反相载波。

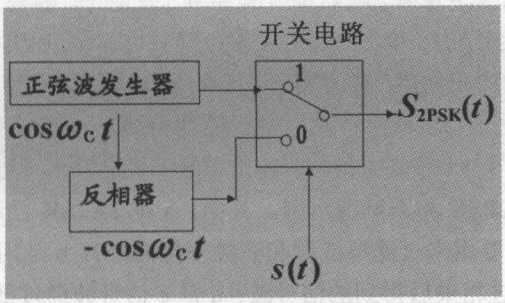


图 4-40 键控调相法调制模型

如图 4-41 所示是一个二进制数字基带信号 1011001 的 2APSK 调制示例，其中 $s(t)$是数字基带信号的矩形脉冲波形，$c(t)$是一个正弦载波波形，2APSK 是经过相位调制后的已调波形。从中可以看出，当信码为 1 时，2APSK 波形的相位与载波一样，而当信码为 0 时，2APSK 的相位与载波相位相反。这种调制原理很好理解。

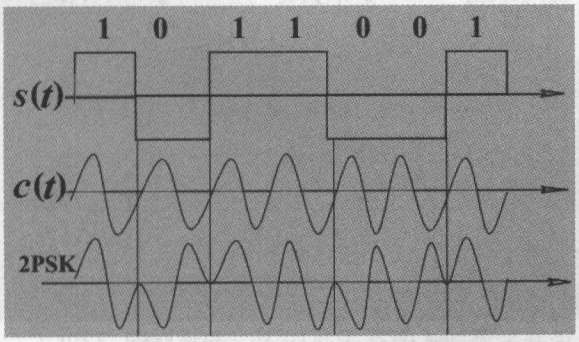


图 4-41 2APSK 调制示例

4.7.8 2APSK 解调原理

2APSK 信号的解调通常都是采用“相干解调法”，其解调模型如图 4-42 所示。在解调过程中需要用到与接收的 2APSK 信号同频、同相的相干载波。抽样判决器是以一个正电平 1 脉冲与低通虑波后的 d 信号进行相乘，d 信号为正电平时输出正脉冲，d 信号为负电平时输出负脉冲。

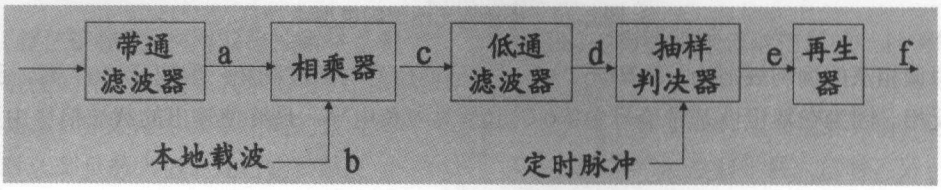


图 4-42　2APSK 相干解调模型

2APSK 的具体解调规则如下：

- 1 码已调信号与载波相乘后，载波输出信号均为正电平，因为 1 码的已调信号与载波相位总是相同，即总为正正相乘或者负负相乘；然后进行幅度相乘。
- 0 码已调信号与载波相乘后，载波输出信号均为负电平，因为 0 码的已调信号与载波相位总是相反，即总是正与负相乘；然后进行幅度相乘。

最终的结果就是已调信号与载波信号同相时输出正电平载波信号，反相时输出负电平载波信号。

如图 4-43 所示是一个 2PSK 相干解调示例。其中，a 是 2APSK 已调信号经过带通滤波器后得到的信号波（带通滤波器主要用来过滤掉低频和高频干扰成分）；b 是用来与 2APSK 信号相乘的本地相干载波；c 是 a 与 b 经过相乘后得到的信号波；d 是 c 信号波经过低通滤波器（去除高频成分）后得到的信号波；e 是 d 信号经过数字取样后得到的离散矩形脉冲信号；f 经过再生器放大后得到的数字脉冲信号，这就是原始的基带数字信号。

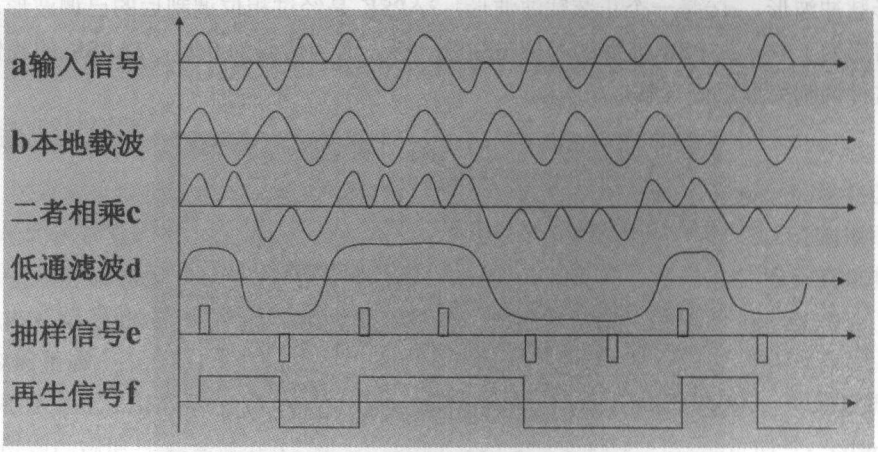


图 4-43　2APSK 相干解调示例

“相干解调”需要一个与发送端调制时所用的载波同频同相的本地载波。但此载波是由接收端的载波提取电路提取的。这里出现一个问题，如果两端载波存在频率和相位偏差，就可能出现解调错误，如原来所有的“1”码都变成了“0”码，所有的“0”码都变成了“1”码。故多数情况下使用本章后面将要介绍的“2DPSK 调制与解调”方法。

如图 4-44 所示，左、右图分别是采用与发送端载波相位相同（相差为 0）、相反（相差为 180 度）的本地载波 2PSK 调制结果，从中可以看出，它们的结果（f 波形）是完全相反的。

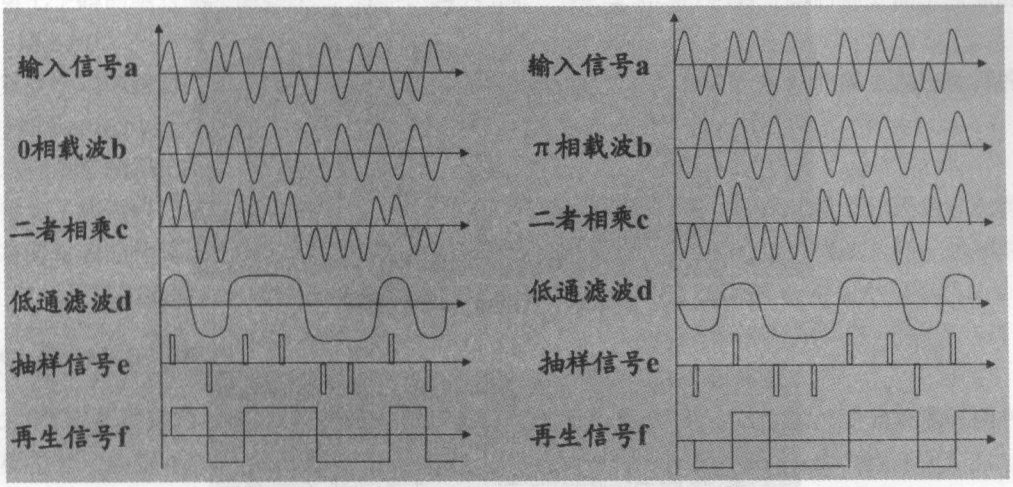


图 4-44 采用 0 相差和 180 度相差本地载波解调的结果比较

4.7.9 2DPSK 调制原理

为了解决“绝对相位调制”中的相位倒置问题，在进行数字调相之前先进行差分编码，再对差分码进行二元数字调相，这就是这里要介绍的“二元差分相位调制”（2DPSK）。

2DPSK 不是利用载波相位的绝对数值传送数字信息，而是用前后码元的相对载波相位值来传送数字信息。所谓“相对载波相位”是指本码元初相位与前一码元初相位之差，即相位偏移，并规定：当相位偏移值为 0 时，输出信码 0；当相位偏移 180 度时，输出信码 1。

2DPSK 已调信号的产生过程是，首先对数字基带信号进行差分编码，即由绝对码变为相对码（差分码），然后再进行绝对调相。2DPSK 的调制原理与 2APSK 的调制原理类似，也是用二进制基带信号作为模拟开关的控制信号轮流选用不同相位的载波来完成 2DPSK 调制，不同的仅是 2DPSK 调制过程中先要对数字基带信号进行差分编码。其调制模型如图 4-45 所示，与如图 4-40 所示的 2APSK 键控调制模型非常相似。

2DPSK 的调制规则与 2APSK 的调制规则一样，即当相对码为 1 时，2DPSK 已调信号波形的相位与载波一样；当相对码为 0 时，2DPSK 已调信号波形的相位与载波相位相反。

如图 4-46 所示是一个 2DPSK 调制示例。图中首先是输入的数字基带信号（绝对码），然后通

过差分变换后得到“相对码”信号波。这里的“相对码”有一个计算规则，即模 2 逻辑加法运算（与异或逻辑运算一样）：$b_n=a_n \oplus b_{n-1}$。式中 b_n 为当前码的相对码，a_n 为当前码的绝对码，b_{n-1} 是 b_n 的前一个码元的相对码，最初的 b_{n-1} 可任意设定。模 2 逻辑加法运算规则是：0+0=0，0+1=1，1+0=1，1+1=0，也就是只要双方值不一样结果才是 1，相同则结果为 0。

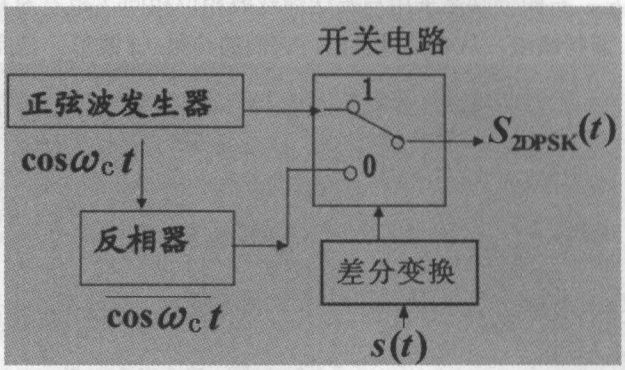


图 4-45　2DPSK 调制模型

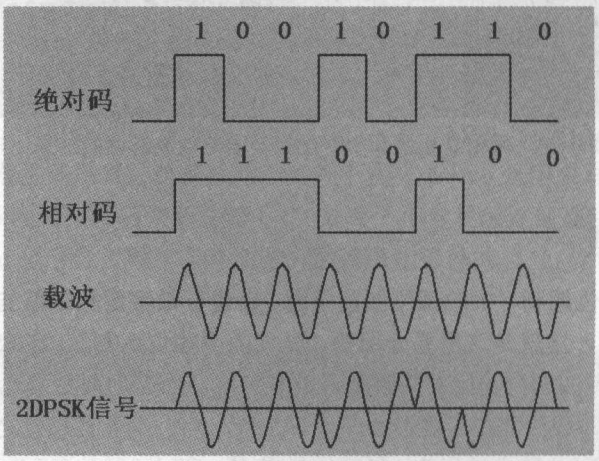


图 4-46　2PDSK 调制示例

下面来分析一下图 4-46 所示的这个示例，假设最初的 b_{n-1} 相对码为 0。

第一个码元的绝对码（a_1）为 1，因为最初的 b_0 相对码为 0，所以它们模 2 加法运算的结果为 1，即 $b_1=1$。

第二个码的绝对码（a_2）为 0，因为 b_1 相对码前面已算出为 1，所以它们模 2 加法运算的结果也为 1，即 $b_2=1$。

第三个码的绝对码（a_3）为 0，因为 b_2 相对码前面已算出为 1，所以它们模 2 加法运算的结果同样为 1，即 $b_3=1$。

第四个码的绝对码（a_4）为 1，因为 b_3 相对码前面已算出为 1，所以它们模 2 加法运算的结果为 0，即 b_4=0。

第五个码的绝对码（a_5）为 0，因为 b_4 相对码前面已算出为 0，所以它们模 2 加法运算的结果为 0，即 b_5=0。

第六个码的绝对码（a_6）为 1，因为 b_5 相对码前面已算出为 0，所以它们模 2 加法运算的结果为 1，即 b_6=1。

第七个码的绝对码（a_7）为 1，因为 b_6 相对码前面已算出为 1，所以它们模 2 加法运算的结果为 0，即 b_7=0。

第八个码的绝对码（a_7）为 0，因为 b_7 相对码前面已算出为 0，所以它们模 2 加法运算的结果为 0，即 b_7=0。

这样最终就得出数字基带信号 10010110 的相对码为 11100100。

得到了相对码，后面载波的变化（也就是 2DPSK 信号的相位）就很容易了，因为它也有一个与相对码对应的计算规则，即：当相对码为“1”时，调制信号的相位与原载波一样；当相对码为“0”时，反转 180 度。这样就很容易画出如图 4-46 所示最后的 2PDSK 波形。

4.7.10 2DPSK 解调原理

2DPSK 解调最常用的方法就是极性比较法（相干解调）和相位比较法（差分解调）两种，其解调模型分别如图 4-47 和图 4-48 所示。

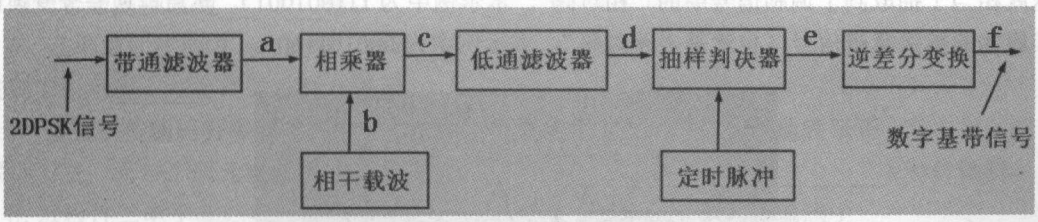


图 4-47 2DPSK 相干解调模型

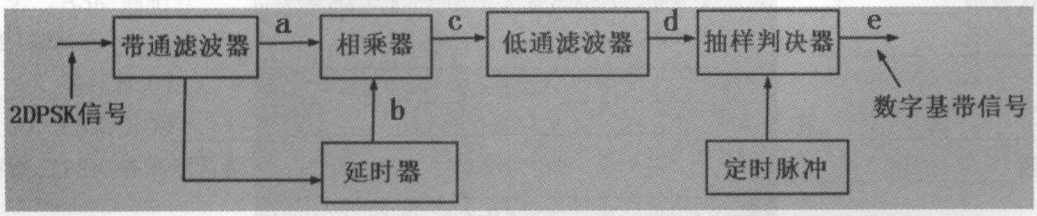


图 4-48 2DPSK 差分解调模型

- a 信号经过与本地相干载波 b 相乘（同相位时为正，不同相位时为负）后得到 c 信号。把一个码元对应的双极性变为单极性。

- c 信号经过低通滤波器过滤高频成分后得到 d 信号。
- d 信号经过定时抽样得到离散的二进制脉冲信号 e（这就相对于抽取到了调制信号中的“相对码”）。
- e 信号经过逆差分变换就得到了原始的数字基带信号 f（也就是“绝对码”）。

2DPSK 解调是 2DPSK 调制的逆过程。通过上节的学习我们已经知道，2DPSK 调制过程中要对绝对码（也就是“数字基带信号”）进行差分处理，转换成相对码，然后再进行绝对相位调制。由此可以得出，2DPSK 解调需要先把 2PDSK 已调信号转换成相对码，然后再把相对码转换成绝对码，这个绝对码就是要解调出来的数字基带信号。

从相对码转换成绝对码的计算也是遵从模 2 逻辑加法运算，运算公式为：$a_n=b_n+b_{n-1}$，a_n 为当前相对码的绝对码，b_n 和 b_{n-1} 分别为当前相对码和前一个相对码。相同为 0，相异为 1。初始相对码任意，通常为 1。

【说明】因为 2DPSK 是靠相邻码元的变化与否来决定“1”码和“0”码的，所以无论接收端用 0 相载波还是 180 度相位相干载波与已调信号相乘，尽管可能得到的波形不同，但 0 码和 1 码对应的波形相位会同步变化，最终再经差分逆变换后，二者得到的结果完全相同。

下面仅以相干解调法进行介绍。

如图 4-49 所示是一个 2DPSK 相干解调示例。图中 a 为 2DPSK 已调信号经过带通滤波器（把高、低频干扰成分过滤掉）得到的 2DPSK 信号。它经过与本地相干载波相乘后得到 c 信号，然后再经过低通滤波器过滤高频成分得到 d 信号；d 信号再经过定时抽样得到离散的二进制脉冲信号 e（这就相当于抽取到了调制信号中的“相对码”，本示例中为 111001001），然后再逆差分变换就得到了原始的数字基带信号（也就是“绝对码”，本示例中为 000101100）。

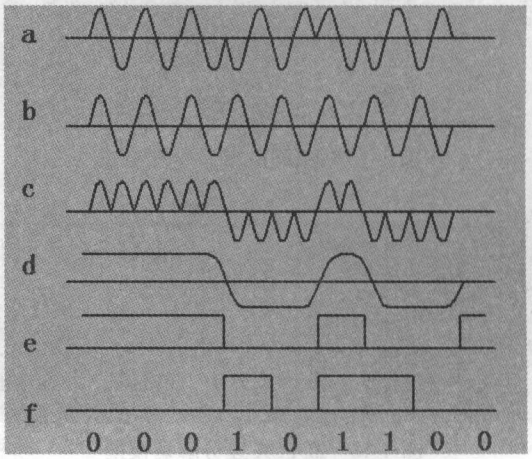


图 4-49　2DPSK 相干解调示例

这里从相对码转换成绝对码的计算也是遵从模 2 逻辑加法运算的，运算公式为：$a_n=b_n \oplus b_{n-1}$，b_n 和 b_{n-1} 分别为当前相对码的绝对码以及前一个相对码的绝对码。计算方法参见前面介绍的 2DPSK

调制中的绝对码与相对码的转换，大家可以自己计算验证一下。

4.8 课后自测题

为了巩固本章所学的内容，请认真做下面的自测题，参考答案见附录。

4.8.1 填空题

1．数据通信基本模型包括_____、_____和_____三大部分，交换机、路由器设备属于_____。

2．计算机网络数据传输方式分为两种，按比特位一位位地传输称为_____，整字节一次性传输称为_____；同步传输是以_____为传输单位，主要分为_____、_____和_____3种，异步传输是以_____为传输单位。

3．奈奎斯特第一准则描述了_____，从中可以获知，每赫兹带宽的理想低通信道的最高码元传输速率是每秒_____个码元，其计算公式为_____。

4．香农公式指出了_____，其计算公式为_____。

5．采样定理包括_____和_____两个基本的过程，从时域采样定理中可以得知，当采样频率大于或等于信号中最高频率的_____倍时才能保持信号的不失真。一般实际应用中保证采样频率为信号最高频率的_____倍。

6．在单极性波形中，二进制1表示_____，二进制0表示_____。而在双极性波形中，二进制1表示_____，二进制0表示_____。

7．常见的传输码有多种，其中_____的编码规则是："1"码交替地表示正、负电平，而"0"码保持不变；而一个码元被转换成两个码元的有_____、_____和_____3种传输码。

8．双极性二电平非归零码中的"二电平"是指_____，在常见的传输码中_____、_____和_____都属于双极性二电平非归零码。

9．2ASK调制有_____和_____两种方法，2ASK解调有_____和_____两种方法。2FSK调制有_____和_____两种方法，2FSK解调主要有_____、_____、_____和_____等几种方法。

10．PSK分为_____和_____两种，通常采用_____。PSK有_____和_____调制方法。2PSK解调通常都是采用_____。2DPSK解调最常用的方法是_____和_____两种。

4.8.2 选择题（可多选）

1．下面关于"基带传输"说法正确的是（　　）。

A．基带传输只适用于数字信号传输

B．基带传输中的信号频率不会被改变

C．基带传输中的数字信号是直接在传输介质上传输的

D．基带传输适用于远距离传输

2．下面关于“频带传输”说法正确的是（　　）。

A．频带传输只适用于模拟信号传输

B．频带传输中原始信号必须经过频率调制

C．频带传输中原始信号必须经过编码

D．频带传输适用于远距离传输

3．在数字基带传输或频带传输中，在接收端都要进行抽样判决的原因是（　　）。

A．发送端发送的是模拟信号

B．原始数字信号经过编码或调制后成了模拟信号

C．因为经过滤波器过滤后仍可能存在连续波形的干扰信号

D．因为经过滤波器过滤后仍可能存在高频信号

4．下列技术中不是采用串行传输方式的是（　　）。

A．USB　　B．ATA

C．SATA　　D．RS-232C

5．下列关于同步传输与异步传输说法正确的是（　　）。

A．同步传输的有效传输率高于异步传输

B．同步传输的实现比异步传输简单

C．同步传输一次传输的数据量小于异步传输

D．HDLC 是采用同步传输模式，ATM 是采用异步传输模式

6．下面关于单双工模式说法正确的是（　　）。

A．单工模式在同一时刻只允许传输一个方向的数据，不同时刻可以传输不同方向的数据

B．在单工模式下，一方只能发送数据，另一方只能接收数据

C．在半双工模式下，一方只能发送数据，另一方只能接收数据

D．在全双工模式下，双方都可以同时发送和接收数据

7．下面关于奈奎斯特第一准则理解不正确的是（　　）。

A．公式 MaxRB=2W 同时适用于理想低通信道和理想带通信道

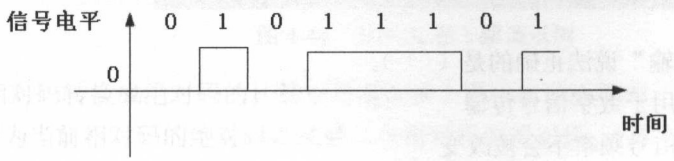
B．公式 MaxRB=2W 只适用于没有干扰情况下的基带传输

C．在理想低通信道中，每赫兹带宽的最高码元传输速率是每秒 1 个码元

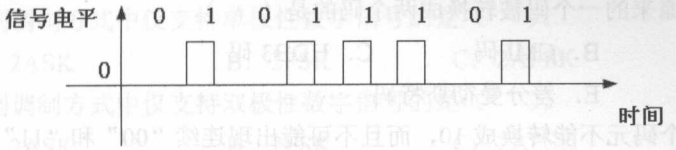
D．在理想带通矩形信道中，每赫兹带宽的最高码元传输速率是每秒 2 个码元

8．以下信号波形属于归零码的是（　　）。

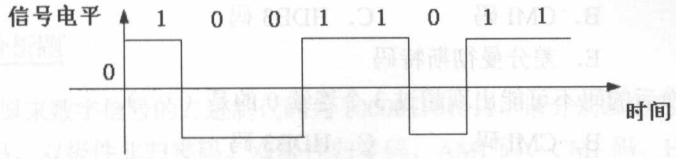
A．



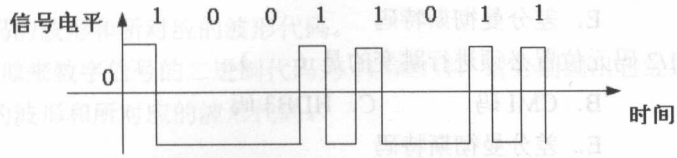
B.



C.

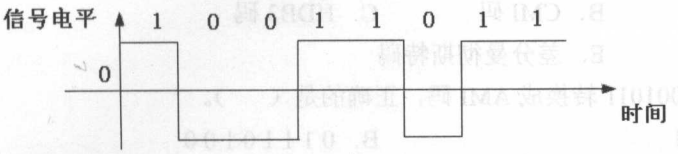


D.

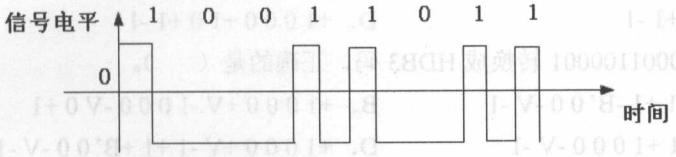


9. 下面关于 10011011 的双极性归零码正确的是（　　）。

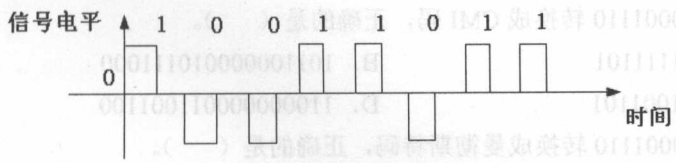
A.



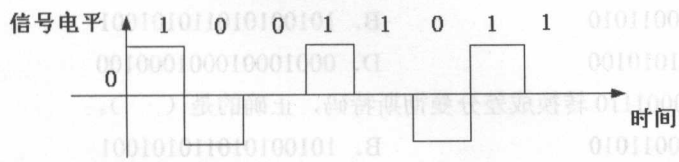
B.



C.



D.



10. 下列传输码中是单极性非归零码变形的是（　　）。

A. AMI 码　　B. CMI 码　　C. HDB3 码

D. 曼彻斯特码　　E. 差分曼彻斯特码

11. 下列传输码中是双极性非归零码变形的是（　　）。

A. AMI 码　　B. CMI 码　　C. HDB3 码

D. 曼彻斯特码　　E. 差分曼彻斯特码

12. 下列传输码中是原来的一个码被转换成两个码的是（　　）。

A. AMI 码　　B. CMI 码　　C. HDB3 码

D. 曼彻斯特码　　E. 差分曼彻斯特码

13. 下列传输码中一个码元不能转换成 10，而且不可能出现连续“00”和“11”的是（　　）。

A. AMI 码　　B. CMI 码　　C. HDB3 码

D. 曼彻斯特码　　E. 差分曼彻斯特码

14. 下列传输码中转换后的码不可能出现超过 3 个连续 0 的是（　　）。

A. AMI 码　　B. CMI 码　　C. HDB3 码

D. 曼彻斯特码　　E. 差分曼彻斯特码

15. 下列传输码中在 1/2 码元位置必须进行跳变的是（　　）。

A. AMI 码　　B. CMI 码　　C. HDB3 码

D. 曼彻斯特码　　E. 差分曼彻斯特码

16. 下列传输码中 1 码必须用“10”转换，0 码必须用“01”转换的是（　　）。

A. AMI 码　　B. CMI 码　　C. HDB3 码

D. 曼彻斯特码　　E. 差分曼彻斯特码

17. 二进制信号码 10001011 转换成 AMI 码，正确的是（　　）。

A. 1 0 0 0 1 0 1 1　　B. 0 1 1 1 0 1 0 0

C. +1 0 0 0 -1 0 +1 -1　　D. +1 0 0 0 +1 0 +1 -1

18. 二进制信号码 100001100001 转换成 HDB3 码，正确的是（　　）。

A. +1 0 0 0 +V -1 +1 -B' 0 0 -V -1　　B. +1 0 0 0 +V -1 0 0 0 -V 0 +1

C. +1 0 0 0 +V -1 +1 0 0 0 -V -1　　D. +1 0 0 0 +V -1 +1 +B' 0 0 -V -1

19. 二进制信号码 110001110 转换成 CMI 码，正确的是（　　）。

A. 111101010111111101　　B. 101100000010111000

C. 110001010111001101　　D. 110000000011001100

20. 二进制信号码 110001110 转换成曼彻斯特码，正确的是（　　）。

A. 1001010101100110 10　　B. 1010010101101010 01

C. 0101000000010101 00　　D. 0001000100010001 00

21. 二进制信号码 110001110 转换成差分曼彻斯特码，正确的是（　　）。

A. 1001010101100110 10　　B. 1010010101101010 01

C. 0101000000010101 00　　D. 0001000100010001 00

22. 下列调制技术中属于多载波调制的是（　　）。

A. ASK　　B. FSK　　C. PSK

D. FDM　　E. OFDM

23. 下列光波调制技术中建立在抽样定理基础上的是（　　）。

A. FDM　　B. OFDM　　C. TDM　　D. WDM

24. 下列调制方式中仅支持单极性数字信号的是（ ）。

A. 2ASK　　B. 2FSK　　C. 2APSK　　D. 2DPSK

25. 下列调制方式中仅支持双极性数字信号的是（ ）。

A. 2ASK　　B. 2FSK　　C. 2APSK　　D. 2DPSK

4.8.3 综合分析题

1. 假设原来数字信号的二进制代码为100001101011，请分别画出它所对应的单极性非归零码、单极性归零码、双极性非归零码、双极性归零码、AMI码、CMI码、HDB3码、曼彻斯特码和差分曼彻斯特码的波形和所对应的波形代码。

2. 假设原来数字信号的二进制代码为11010111，请分别画出它经过2ASK、2FSK和2APSK相干调制后的波形和所对应的波形代码。

5 物理层

从本章开始就要正式介绍计算机网络体系结构中各层的主要功能与特性了。本章介绍的是体系结构中的最低层——物理层。就像我们要构建现实中的交通网络先要进行最基本的基础设施建设一样，先来为网络“修路”。

本章首先对物理层的主要功能、所定义的四方面特性进行介绍，然后着重介绍在计算机网络中常见的一些传输介质、连接器及相关标准，以及在物理层中实现物理传输信道复用的几种技术原理，最后介绍已经出现的几种典型 WLAN 接入规范，特别要注意对目前典型应用的 IEEE 802.11n/ac/ad 这 3 种规范的了解。

本章重点与难点：

- 物理层所定义的电气特性
- EIA/TIA 568A 和 EIA/TIA 568B 标准
- 直通双绞线和交叉双绞线在线序排列和应用方面的区别
- 单模和多模光纤的定义及主要特性
- 主要光纤模块和连接器类型
- 尾纤和跳纤的主要特点和主要应用
- 频分复用、时分复用和波分复用原理
- IEEE 802.11n/ac/ad 三种规范的主要特性

5.1 物理层基础

通过前面各章的学习已经知道，物理层是计算机网络体系的最低层，负责为数据通信构建一条

连通的、电气化的物理链路或者传输信道。工作在物理层的设施包括：各种传输介质、集线器（Hub）、中继器、无线 AP 等。当然在各种网络设备接口上同样也体现了物理层上的功能，包括各种接口的外观、结构、电气和功能等特性。本节先从宏观角度介绍物理层的主要功能、所定义的四方面特性，以及总体的传输介质分类。

5.1.1 物理层主要功能

虽然物理层的主要目的是构建物理链路，但从物理层的设计考虑方面来说，其功能又有多个方面，具体如下：

（1）构建物理链路。

物理层最基本的功能就是构建用于在计算机网络中进行数据传输的物理链路。它可以只是一段物理介质的连接，也可以是多段物理介质串接而成的，当然必须实现“电气”意义上的真正连接，而不是简单的物理连接。电气化的物理链路建立不仅需要各种传输介质、网络接口、网络设备，还需要有对应的通信协议支持。

（2）透明传输。

由于不同类型计算机网络所使用的传输介质类型不同，在局域网中就有同轴电缆、双绞线、光纤，在广域网中又有各种同步、异步电缆，而且这些传输介质所使用的通信协议和标准各不相同，这就决定了不同的计算机网络，甚至在同一个计算机网络的不同物理链路段都可能有不同的“路”。

物理层除了要把这些不同的“路”修好，还要确保这些不同的“路”能真正在电气意义上连通起来，形成通路，这就需要物理层的“透明传输”功能。物理层的透明传输功能具有屏蔽不同传输介质类型和通信协议的能力，让进行网络通信的各方只看到有“路”可行，而不管修这些“路”所用的具体传输介质和相关的通信协议、标准。

（3）传输数据。

无论是从网络体系结构中哪层发起的通信，最终的数据还得通过最低的物理层传输出去，因为这是网络通信的唯一物理通道。物理层的传输单位是比特（bit，也就是“位”），负责在发送端通过网络接口和传输介质将数据按比特流的顺序一位位地传输到接收端的物理层。

（4）数据编码。

为了实现数据能在物理传输介质上有效、可靠地传输，而且在接收端接收到数据后能根据所接收的比特流特性正确识别在发送端数据链路层发送的数据帧中所具有的帧边界属性，还需要根据不同传输介质的特性对所传输的数据进行对应的处理。这就是物理层的数据编码功能，如归零码、非归零码、曼彻斯特码、差分曼彻斯特码等，参见第 4 章中的介绍。

（5）数据传输管理。

物理层还具有一定的数据传输管理功能，如基于比特流的流量控制、差错控制、物理链路的激活和释放等。

5.1.2 机械特性

总体来说，“物理层”的主要任务是定义与传输介质、连接器及其接口相关的机械特性、电气特性、功能特性和规程（也就是我们通常所说的“协议”）特性这 4 个方面。本节先来介绍第一个特性——机械特性，其他 3 个特性将在后面的各小节中介绍。

物理层的“机械特性”定义了传输介质接线器、网络接口的形状和尺寸、芯线数目和排列顺序，以及连接器与接口之间的固定或锁定装置。

在计算机网络中，我们常见的传输介质是各种同轴电缆、双绞线、光纤等，其实还有经常用于路由器 S 口（串口）连接以及以前各种 Modem 与计算机连接时所使用的各种同、异步串行电缆。不管是哪种传输介质，在进行网络连接时都需要两个方面：一是安装了连接器的网线，另一个是在设备或网卡中的对应网络接口。

不同类型的传输介质连接器、网络接口在电气特性、外观形状以及固定或锁定装上都存在比较大的区别。如图 5-1 所示是细同轴电缆所用的连接器（称为 BNC）和网络接口，而图 5-2 所示是我们常用的以太网双绞线连接器（称为“水晶头”）和 RJ-45 以太网接口。

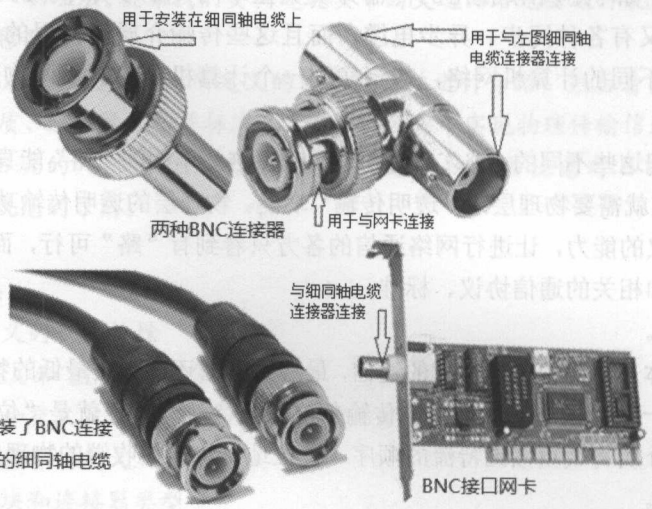


图 5-1　细同轴电缆 BNC 连接器、细同轴电缆网线和 BNC 网卡

目前，光纤传输介质的连接器和对应的网络接口类型比较多，如图 5-3 所示是 LC 类型连接器和其对应的 LC 类型光纤模块网络接口。有关其他类型的光纤连接器和接口将在本章后面介绍。

串行接口方面也有许多连接器和接口类型，主要是里面包括的芯线条数不同（当然里面每条芯线的功能和电气特性肯定也有所区别），但在串行电缆上所安装的连接器都是公、母配对使用的。其中“公头”是针状的，称为 DCE（Data Communications Equipment，数据通信设备），“母头”是孔状的，称为 DTE（Data Terminal Equipment，数据终端设备）。如图 5-4 所示是 DB-25 和 DB-9 两

种连接器，在一条电缆上都会一端是 DTE 连接器，另一端是 DCE 连接器。

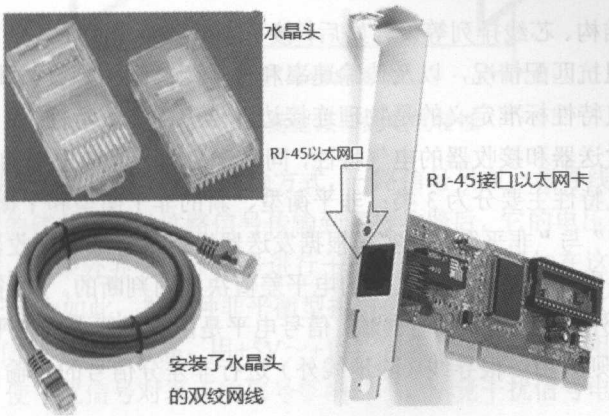


图 5-2　双绞网线连接器、双绞网线和双绞线以太网网卡

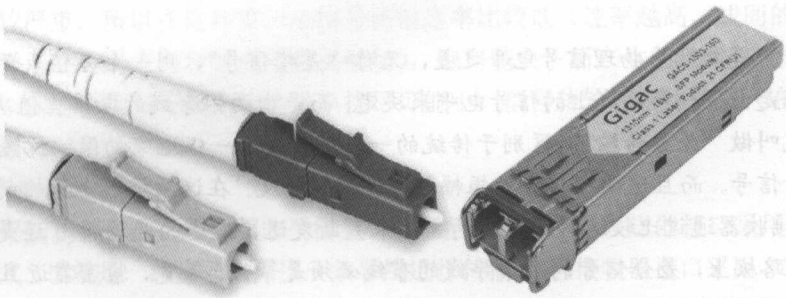


图 5-3　LC 光纤接口连接器和 LC 光纤模块接口

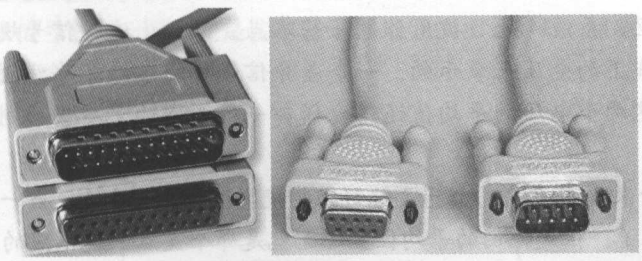


图 5-4　DB-25 和 DB-9 连接器

DTE 端连接的是具有一定数据处理能力和数据收发能力的设备，如 PC 机，DCE 端连接的是在 DTE 和传输线路之间提供信号变换和编码功能并负责建立、保持和释放链路连接的设备，如各种 Modem。路由器一般是同时作为 DTE 和 DCE，担当双重角色，当然具体角色是由路由器上不同的接口担当的。DCE 设备需要与 DTE 设备对接。

有关一些主要类型传输介质连接器的芯线安排及各自的电气特性将在本章后面介绍。

5.1.3 电气特性

网络接口的外观结构、芯线排列等设计好后，接下来就要确定其中各条芯线在传输信号时线路上信号电压的高低、阻抗匹配情况，以及传输速率和传输距离限制等参数属性。

早期的物理层电气特性标准定义的是物理连接边界点上的电气特性，而目前使用的新电气特性标准定义的都是接口发送器和接收器的电气特性，同时还给出了互连电缆的相关规程。目前所使用的新的物理层接口电气特性主要分为 3 类：非平衡型、新的非平衡型和平衡型。

这里所说的“平衡”与“非平衡”其实是根据发送器发送信号或者接收器接收信号时信号电平是由单根线的电平值决定还是由两根线之间的电平差值决定而判断的。“平衡”传输模式中都是采用一对线（除了地线之外）进行差分信号传输，信号电平是由两根线上的电平差决定的，如双绞线；而“非平衡”传输模式则只用一根导线（除地线外）进行非差分信号的传输，信号电平仅由一根信号线上的电平决定。



“差分信号”是指两个物理信号电平之差，又称“差模信号”，用一个差值表示。也就是指最终的信号电平不是由某一条导线上的信号电平来决定，而是由两根导线的电平差值决定的。使用差分信号的传输就叫做“差分传输”，区别于传统的一根信号线、一根地线的做法。差分传输是在两根线上同时传输信号，而且这两个信号的振幅相等、相位相反。在这两根线上传输的信号就是“差分信号”。信号接收器通过比较这两个电压的差值来判断发送器发送的是逻辑 0 还是逻辑 1。但是需要注意，在电路板上，差分信号的两条导线的布线必须是等长、等宽、紧密靠近且在同一层面的两根线。只有这样才可能做到尽可能使这两根导线上的差分信号同幅、反相、同步传输。

从严格意义上来讲，所有电压信号都是差分的，因为一个电压只能是相对于另一个电压而言的。在某些系统里，系统“地”被用作电压基准点。此时，这种信号规划被称为单端的，因为信号是用单条导线上的电压来表示的。一个差分信号作用在两条导线上，信号值是两条导线间的电压差，而且这两个电压的平均值还是会保持一致，因为干扰会同时作用于两条导线上，并且效果一样。

我们用一个比喻来说明，“差分信号”就好比是跷跷板上的两个人，当一个人被跷上去的时候，另一个人就被跷下来了，但是他们的平均位置（也就是那个支点）是不变的。这就是“差分信号”传输的原理和主要优势，因为这种反相、等幅差分信号传输方式可以有效地抑制外界干扰，具体原理将在后面的“平衡型”接口标准中介绍。

1. 非平衡型

“非平衡型”接口一般是指早期采用分立元件技术设计的，其信号发送器和接收器均采用非平衡方式工作。即每路信号仅使用一根导线传输（是“单线传输方式”），然后所有信号共用一根信号地线（发送器和接收器的地线是相连的，形成统一的接地），如图 5-5 所示。

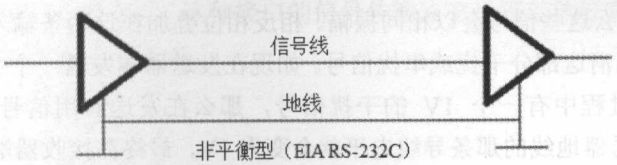


图 5-5 非平衡型接口的电气特性

这种非平衡型接口中采用的是非差分传输方式，一路信号只在一根导线上传输，也就没有另一条导线上的等幅、反相的信号，所以这路信号传输到了接收器后，它的电压也只由这条导线上的信号电平决定。这样一来，如果外界有一个信号进行干扰，就会直接叠加在这条导线的信号上，导致最终的信号失真。也正因为如此，在这种非平衡型接口标准（典型代表为 CCITT V.28 建议的 EIA RS-232C）中所采用的信号电平比较高，用+5V～+15V 表示二进制“0”，用-5V～-15V 表示二进制“1”。其目的就是为了使干扰信号对真正的信号影响较小，毕竟干扰信号电平通常不会很高。如果信号电平比较低的话，干扰信号的影响就会更严重了。

另外，同样由于使用的是单线传输，没有一个用来抵消各种干扰和串扰的线对，不同导线上的信号串扰也比较严重，所以在这种接口中信号传输速率比较低（速率越高，线间的串扰越大）。就像我们跑步一样（其实用汽车行驶来形容更贴切），跑得越快，所产生的风对其他相邻跑道上的运动员干扰越大。为了使不同导线上的信号串扰尽可能小，信号传输速率比较低（在 20Kb/s 以内），传输距离比较短（15 米以内）。

2. 新的非平衡型

“新的非平衡型”标准是原来“非平衡型”标准的改进版本，发送器接口采用的是上面介绍的非平衡工作方式（也就是一路信号也只用一条导线进行传输），但接收器接口采用“差分”工作方式，而且发送器和接收器不共用一条地线，如图 5-6 所示，主要用于发送端和接收端固定的应用场景。

在这种接口中，发送器接口仍使用一根导线发送信号，但在接收器端是两条导线输入的，除了发送器直接驱动的那条信号线外，还有就是接自发送器地线的那条导线，接收信号的最终电平是由这两条导线上的差值来决定的。

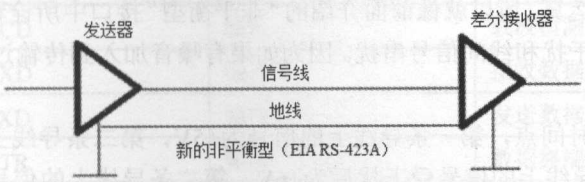


图 5-6 新的非平衡型接口的电气特性

在这种新的非平衡型接口标准（典型代表就是 CCITT V.10/X.26 建议的 EIA RS-423A）中，发送器采用单线驱动，接收器采用双端输入，接收器最终信号的电平就是这两条线的差值（进行减法运算），所以称为“差分接收器”。

在这种情况下，如果没有干扰和串扰，接收器的最终信号就是发送器发送过来的那个信号电平；

如果有干扰或串扰，那么这些信号会以相同振幅、相反相位叠加在这两条输入导线上，最终在接收器的差分结果就可以抵消这部分干扰或串扰信号。如现在发送器端发送一个 5V 电平的信号到接收器端，而同时在传输过程中有一个 1V 的干扰信号，那么在发送有用信号的那条导线上的就是 5+1=6V，而在接到发送器地线的那条导线电平就会变为 1V，最终在接收器端经过减法运算（因为要求它们的电平差），最终的信号电平就仍为 6-1=5V，保持不变。

因为在接收器端采用了差分工作方式，可抵消外界干扰或线间串扰，所以它的信号电平可以比较低（这对电路中的芯片都是非常有利的），使用+4V～+6V 表示二进制“0”，用-4V～-6V 表示二进制“1”。同样传输速率也可以比较高，在 10 米以内的近距离情况下，传输速率可达 300Kb/s，但当传输距离达到 1000 米时，信号传输速率只有 3Kb/s 以下。

3. 平衡型

“平衡型”接口是目前最广泛采用的一种物理层接口（如常见的 RJ-45 接口），一般是采用集成电路技术设计的。它规定“发送器”采用双线平衡发送方式，而“接收器”采用差分处理方式，也不共用地线，如图 5-7 所示。当然，这里的发送器和接收器是相对的，也就是在双向传输中两端是互为发送器、接收器的。

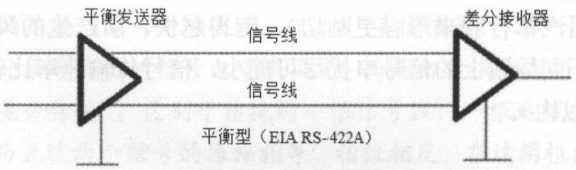


图 5-7 平衡型接口的电气特性

在平衡型接口标准（典型代表为 CCITT V.11/X.27 RS-422A）中规定，两条导线承载同样的传输，但它们是等幅、反相的，反相信号看上去就像源信号在镜子中的映像。接收端并不侦听任何一条信道中的信号，而是检测两条信道中信号的差值。当两个反相信号抵达接收端时需要进行减法运算，因为互为反相，所以两者相减的结果就是第一个信号值的两倍。比如，在某个特定时间点，第一个信号为+5V，那么第二个信号就是-5V，两者相减 5-(-5)=10V，相当于放大了原始信号。同样，由于接收器采用了差分运算，所以就像前面介绍的“非平衡型”接口中所诠释的差分运算原理一样，它也可以抵消外界信号干扰和线间信号串扰。因为如果有噪音加入到传输过程中，那么噪音将以同样的方式影响两个信号。

比如，在某个特定时间点，第一条导线上的信号为+5V，第二条导线上的信号就是-5V，假设噪音为+1V，则第一条导线上的信号受干扰后为+6V，第二条导线上的信号受干扰后为-4V。如果噪音为-1V，则第一条导线上的信号受干扰后为+4V，第二条导线上的信号受干扰后为-6V。这时候接收端取两个信号的差值。但在这里要特别注意的是，不论噪音是正还是负，不论噪音有多大，在相减的过程中都被抵消了，出来的差值都是 10V。就是这种抵消的能力，使得平衡型传输获得较高的传输速率，同时具有较高的抗干扰能力。

基于上述原因，在平衡型接口标准中也是使用较低的信号电平，如用+4V～+6V 表示二进制

“0”，用-4V～-6V 表示二进制“1”。这种接口的信号传输速率也可以比较高，当在 10 米以内的近距离传输时，速率可达 10Mb/s 或更高。

5.1.4 功能特性

物理层的“功能特性”是指明传输介质中各条线上所出现的某一电平的含义以及物理接口各条信号线的用途，包括：接口信号线的功能规定（如哪些是信号线、控制线、时钟信号线）和接口信号线的功能分类。

表 5-1 所示是在 10Base-T/100Base-TX 和 1000Base-T 这两大类以太网标准中 MDI（Medium Dependent Interface，介质相关接口）类型网络接口或其对应的双绞线水晶头中 8 个引脚的功能定义。

表 5-1 双绞线 MDI 接口引脚功能定义

接口引脚序号	10Base-T/100Base-TX		1000Base-T	
	信号缩写符	功能说明	信号缩写符	功能说明
1	Tx+	发送数据	BIDA+	双向数据线 A+
2	Tx-	发送数据	BIDA-	双向数据线 A-
3	Rx+	接收数据	BIDB+	双向数据线 B+
4	保留	—	BIDC+	双向数据线 C+
5	保留	—	BIDC-	双向数据线 C-
6	Rx-	接收数据	BIDB-	双向数据线 B-
7	保留	—	BIDD+	双向数据线 D+
8	保留	—	BIDD-	双向数据线 D-

表 5-2 所示是 RS-232 串行接口中 9 个引脚的功能定义。

表 5-2 串行接口引脚功能定义

接口引脚序号	信号缩写符	信号方向	功能说明
1	DCD	输入	载波检测
2	RXD	输入	接收数据
3	TXD	输出	发送数据
4	DTR	输出	数据终端准备好
5	GND	公共端	信号地
6	DSR	输入	数据装置准备好
7	RTS	输出	请示发送
8	CTS	输入	清除发送
9	RI	输入	振铃指示

CCITT V.24 建议采用每根信号线定义一个功能的方法，是规定接口信号线功能的主要标准之一。而 CCITT X.24 建议采用每根接口信号线可定义多个功能的方法，这种多重复用一根接口信号线的方法可以减少接口信号线的数量。接口信号线按功能一般可分为数据信号线、控制信号线、定时信号线和接地线 4 类。信号线的名称可以采用数字、字母组合或英文缩写 3 种方式来命名。CCITT V.24 建议用数字命名法。

按 CCITT V.24 建议的接口信号线命名方法，DTE-DCE 接口信号线名称的第一位均为“1”，称为 100 系列接口信号线。相应地，CCITT V.24 建议用于 DTE-ACE（Automatic Calling Equipment，自动呼叫设备）设备的连接接口信号线名称的第一位均为“0”，称为 200 系列接口信号线。其他还有一些常用的接口标准，如 X.25（分组型公用数据网 DTE-DCE 接口标准）、X.20（公用数据网起止式传输业务的 DTE-DCE 接口标准）、X.20 bis（连接在公用数据网上的 V 系列建议起止式传输 DTE-DCE 兼容性接口标准）等。

5.1.5 规程特性

物理层的“规程特性”指明利用接口传输比特流的全过程及各项用于传输的事件发生的合法顺序，包括事件的执行顺序和数据传输方式，即在物理连接建立、维持和交换信息时，DTE/DCE 双方在各自电路上的动作序列。

每种计算机网络都有对应的物理层规程（也就是通信协议），如我们常见的以太网对应的物理层规程就是 IEEE 802.3 系列标准，而 WLAN 中对应的物理层规程就是 IEEE 802.11 系列接入规程。目前在广域网中，由 CCITT 建议在物理层使用的规程有 V.24、V.25、V.54 等 V 系列标准，以及 X.20、X.20 bis、X.21、X.21 bis 等 X 系列标准。

有关这些标准的具体规程特性对我们从事网络职业的人士来说可以不去了解，因为它主要对从事网络产品开发的人员有意义。

5.1.6 传输介质分类

在计算机网络中，“传输介质”通俗地讲就是各种网线，用于连接各种网络设备。当然不同类型的计算机网络，它们所采用的传输介质类型不一样，如目前在以太局域网中最常用的是双绞线，早期的计算机网络还有用同轴电缆的。以上仅是针对有线网络而言，现在的无线计算机网络技术发展也相当快，它们也有对应的传输介质。

以上仅是大的分类，其实在像双绞线、光纤、同轴电缆这些大类中又有许多小的分类。但不管怎么样，还是可以根据这些传输介质的导向性分为以下两大类：

- 导向性传输介质：也就是信号被固定沿着信道的一个或多个方向进行传输的介质，如双绞线、同轴电缆和光纤等。导向性传输介质其实就是有线网络中的传输介质，是看得见、摸得着的。在同轴电缆、双绞线中传输的是电信号，在光纤中传输的是光信号。
- 非导向性传输介质：也就是信号在信道中的传输没有固定方向的传输介质，如空气中的无线电波（如红外线、紫外线等）、电磁波，通常是特指在各种无线网络（如 WLAN、卫

星通信）中所使用的传输介质。非导向性传输介质其实就是无线网络中的传输介质，是看不见、摸不着的，传输的信号是电波、电磁波等。

本章仅介绍导向性传输介质，非导向性传输介质是各种电磁波，如 WLAN 网络、卫星通信、蓝牙数据通信中的传输介质都是电磁波。

5.2 双绞线

双绞线（Twisted Pair）在以太局域网连接中应用最为广泛，主要得益于它的廉价性和快速发展、提高的技术和性能。以太网双绞线的最大传输距离为 100m。

双绞线是采用我们前面介绍的平衡型电气特性的，所以每一路信号的传输需要使用两根芯线。之所以称之为“双绞线”就是这两条传输同一路信号（相位相反）的芯线不是平行的，而是按一定规则（绞合度）绞合在一起，如图 5-8 所示。



图 5-8 双绞线的绞合形式

为了实现双向传输，所以在双绞线中至少需要 2 对、4 根芯线。又为了提高传输速率，实现不同类型双绞线的总体传输速率需求，所以在一条双绞线电缆中就包括了 4 对、8 根芯线。

5.2.1 双绞线的分类

以太网双绞线（Twisted-Pair Cable）主要用于传输模拟信号，但也适用于数字信号的传输，特别适用于较短距离的信息传输，是目前局域网中常用的传输介质。但双绞线最早是在电话交换系统中使用的，后来才移植到计算机网络中，当然对应的材质和标准都不一样。

双绞线有两个大的分类标准。根据是否带有屏蔽层，分为“屏蔽双绞线”（Shielded Twisted Pair，STP）和“非屏蔽双绞线”（Unshielded Twisted Pair，UTP）两大类，如图 5-9 所示。

STP 双绞线以金属箔对里面的 8 根芯线进行屏蔽包裹，以减少外界以及芯线间的干扰和串音，屏蔽性能好、搞干扰能力强，主要应用于屏蔽、抗干扰性能要求比较高的场景。

UTP 双绞线则没有 STP 这一层屏蔽用的金属箔。虽然屏蔽和抗干扰性能不如 STP 类型，但对于绝大多数企业局域网来说则影响不大，更为重要的是 UTP 类型双绞线的价格要远比 STP 类型的便宜。

除了分为 STP 和 UTP 外，还可根据双绞线的电气性能来进行分类，分成了 1 类、2 类、3 类、4 类、5 类、超 5（5e）类、6 类、6a 类，一直到当前最新的 7 类。其实这是根据不同时期双绞线的技术标准来分的，类型数字越大，版本越新、技术越先进，带宽也越宽，当然价格也越贵。双绞线技术标准是在美国通信工业协会（TIA）的 EIA/TIA-568A 或 EIA/TIA-568B 标准中定义的。

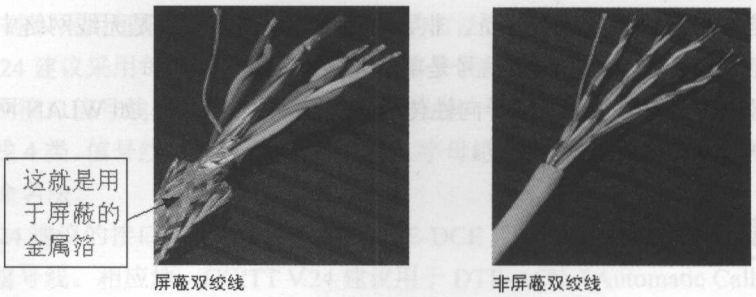


图 5-9　STP 和 UTP 双绞线

- 1 类：是 ANSI/EIA/TIA-568A 标准中最原始的非屏蔽双绞铜线电缆，但它开发之初的目的不是用于计算机网络数据通信，而是用于电话语音通信。
- 2 类：是 ANSI/EIA/TIA-568A 和 ISO 2 类/A 级标准中第一个可用于计算机网络数据传输的非屏蔽双绞线电缆，工作频率为 1MHz，传输速度最高为 4Mb/s。
- 3 类：是 ANSI/EIA/TIA-568A 和 ISO 3 类/B 级标准中专用于 10BaseT 以太网络的非屏蔽双绞线电缆，工作频率为 16MHz，传输速度最高为 10Mb/s。
- 4 类：是 ANSI/EIA/TIA-568A 和 ISO 4 类/C 级标准中用于令牌环网络的非屏蔽双绞线电缆，工作频率为 20MHz，传输速度最高为 16Mb/s。
- 5 类：是 ANSI/EIA/TIA-568A 和 ISO 5 类/D 级标准中用于运行 CDDI（Copper Distributed Data Interface，铜线分布式数据接口）和快速以太网的非屏蔽双绞线电缆，工作频率为 100MHz，传输速度最高为 100Mb/s。CDDI 是基于双绞铜线的 FDDI（Fiber Distributed Data Interface，光纤分布式数据接口）。
- 超 5（5e）类：是 ANSI/EIA/TIA-568B.1 和 ISO 5 类/D 级标准中用于运行快速以太网的非屏蔽双绞线电缆，工作频率也为 100MHz，传输速度最高也为 100Mb/s。与 5 类线缆相比，超 5 类在近端串扰、串扰总和、衰减和信噪比这 4 个主要指标上都有较大的改进。
- 6 类：是 ANSI/EIA/TIA-568B.2 和 ISO 6 类/E 级标准中规定的一种非屏蔽双绞线电缆，主要应用于快速以太网和千兆以太网中。因为它的工作频率可达 200 MHz～250MHz，是超 5 类线带宽的 2 倍，最高传输速度可以达到 1000Mb/s 即 1Gb/s。
- 6a 类：是 6 类线的改进版，同样是 ANSI/EIA/TIA-568B.2 和 ISO 6 类/E 级标准中规定的一种非屏蔽双绞线电缆，主要应用于千兆网络中。工作频率可达 500MHz，最大传输速度可达 10Gb/s，而且在串扰、衰减和信噪比等方面有较大改善。
- 7 类：是 ISO 7 类/F 级标准中最新的一种双绞线，主要是为了适应万兆以太网技术的应用和发展。但它不再是一种非屏蔽双绞线，而是一种屏蔽双绞线，工作频率可达 600MHz，最高传输速率可达 10Gb/s。

【说明】在北美标准中，双绞线及其他一些网络传输介质的标准制作通常是由以下 3 个主要组织进行的：

- ANSI（American National Standard Institute，美国国家标准化组织）
- TIA（Telecommunication Industry Association，电信工业联合会）
- EIA（Engineering Institute Association，工程技术协会）

而且通常情况下，其中的一个组织颁布一个标准，再由另外两个组织进行修正。在欧洲，电缆标准由 CENELEC（欧洲电子技术标准委员会）发布，该组织的成员来自于欧洲 19 个国家的电子技术委员会。ISO/IEC 将采纳 CENELEC 的标准作为 ISO/IEC 的标准。

5.2.2 双绞线连接器

为了方便双绞线中的每根芯线与水晶头的引脚对应，对其中的 8 根芯线染上 4 组不同的颜色，其中 4 根的颜色分别是绿色、蓝色、棕色和橙色，另外 4 根的颜色为以上 4 种颜色与白色的相间色，即白绿、白蓝、白棕和白橙，然后有对应颜色的每两根芯线为一组，按一定规则绞织在一起，即构成了 4 个绞线对。

每条以太网双绞线通过两端安装的 RJ-45 连接器（俗称水晶头）将各种网络设备连接起来。RJ-45 连接器引脚序号如图 5-10 所示。

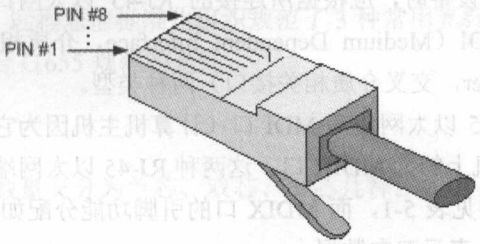


图 5-10 RJ-45 连接器引脚序号示意图

1. 双绞网线制作标准

RJ-45 连接器引脚序号与双绞线中的芯线颜色具有一定的对应关系，EIA/TIA 的布线标准中规定了两种双绞线的线序 568A 和 568B。目前主要使用 EIA/TIA 568B 标准。

- EIA/TIA 568A：白绿－1，绿－2，白橙－3，蓝－4，白蓝－5，橙－6，白棕－7，棕－8。
- EIA/TIA 568B：白橙－1，橙－2，白绿－3，蓝－4，白蓝－5，绿－6，白棕－7，棕－8。

在双绞线的以太网连接中，根据所连接的设备的不同需要使用两类不同线序制作的双绞网线分别是“直通线”（Straight-Through Twisted-Pair Cable）和“交叉线”（Crossover Twisted-Pair Cable）。

在直通线中，双绞线两端水晶头中的线序均为 EIA/TIA 568B 标准，如图 5-11 所示。在交叉线中，一端水晶头中的线序为标准 568B，另一端水晶头中的线序为 EIA/TIA 568A 标准，如图 5-12 所示。这两种双绞线有着不同的设备用途，不能混用。

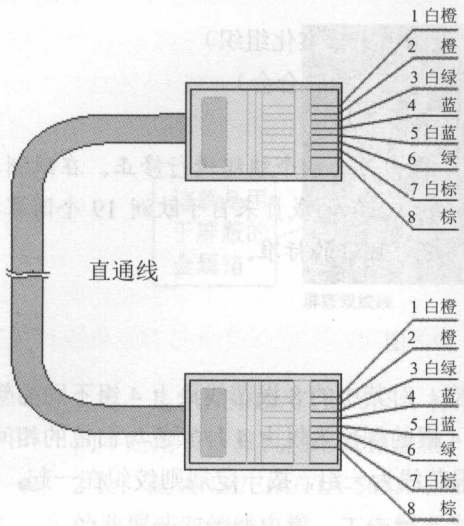


图 5-11 直通线两端线序示意图

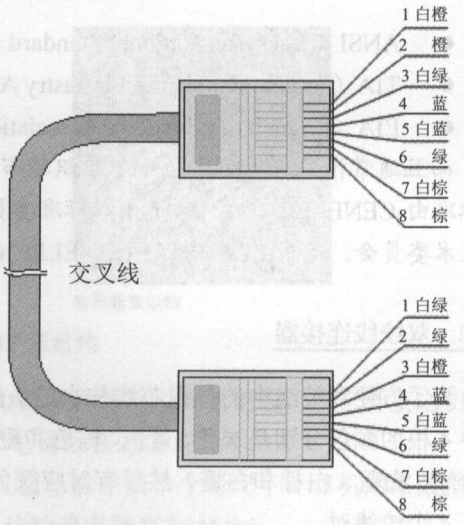


图 5-12 交叉线两端线序示意图

2. 直通线与交叉线使用原则

使用以太网双绞线连接设备时，应根据所连接的 RJ-45 以太网口类型选择以太网双绞线的类型。RJ-45 以太网口分为 MDI（Medium Dependent Interface，介质相关接口）和 MDIX（Medium Dependent Interface Cross-over，交叉介质相关接口）两种类型。

路由器和 PC 上的 RJ-45 以太网口为 MDI 口（计算机主机因为它具备了三层功能，所以与路由器属于同类设备），交换机上的为 MDIX 口。这两种 RJ-45 以太网端口的各引脚功能分配是不一样的，MDI 口的引脚分配参见表 5-1，而 MDIX 口的引脚功能分配如表 5-3 所示。Tx 表示发送数据，Rx 表示接收数据，BID 表示双向数据。

表 5-3 MDIX 口引脚功能分配

端口引脚序号	10Base-T/100Base-TX		1000Base-T	
	信号缩写符	功能说明	信号缩写符	功能说明
1	Rx+	接收数据	BIDB+	双向数据线 B+
2	Rx-	接收数据	BIDB-	双向数据线 B-
3	Tx+	发送数据	BIDA+	双向数据线 A+
4	保留	—	BIDD+	双向数据线 D+
5	保留	—	BIDD-	双向数据线 D-
6	Tx-	发送数据	BIDA-	双向数据线 A-
7	保留	—	BIDC+	双向数据线 C+
8	保留	—	BIDC-	双向数据线 C-

为保证设备正常通信，对于相连的两台设备，一端设备端口的发送数据的引脚需要对应对端设备端口接收数据的引脚。因此，当两端设备都为 MDI 口或 MDIX 口时，需要使用交叉线连接；当一端为 MDI 口一端为 MDIX 口时，需要使用直通线连接。

直通线或交叉线的使用情况可以总结如下：

- 直通线用于连接不同类型设备，比如连接交换机和 PC、交换机和路由器等。
- 交叉线用于连接同种类型设备，比如连接交换机和交换机、路由器和路由器、PC 和 PC 等。

但如果 RJ-45 以太网端口支持 MDI/MDIX 自适应特性，并且启用时端口能自动适应不同线序（自动适应直通线或交叉线），也就是说这时你不管用的是直通线还是交叉线都可以连通的。目前的交换机和路由器端口一般都支持 MDI/MDIX 自适应特性。

5.3 光缆

我们通常所说的“光纤”其实更应该叫“光缆”，因为光纤只是光缆里面真正用于传输数据的部分，而且像双绞线里面有 8 根芯线一样，在一条光缆中也可以包括多条光纤。在光纤的外面还包裹着一层绝缘保护材料。

ITU-T（国际电信联盟）远程通信标准化组织规范了 3 种常用光纤：符合 G.652 规范的光纤、符合 G.653 规范的光纤、符合 G.655 规范的光纤。

5.3.1 光缆的组成

按光缆中所包含的光纤数量又分为单芯、双芯、多芯几种。如图 5-13 所示的上下图就分别对应单芯光缆和多芯光缆结构。

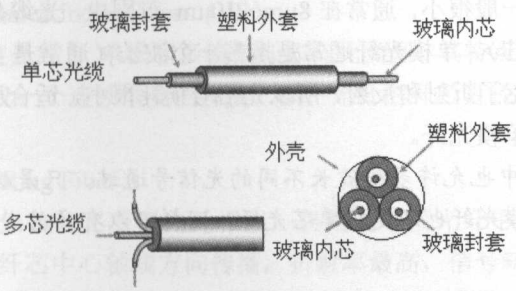


图 5-13 单芯和多芯光缆及其结构

光纤所传递的是光的信号而不再是电的信号，因此光纤传送数据必须有光源。常用的光源有两种：发光二极管（Light Emitting Diode，LED）和激光二极管（Laser Diode）。一条光缆要实现数据传输，必须包括以下 3 个部分：光源、传输介质和检测器。在光纤的一端放上光源，另一端放上检测器，就可以通过光纤进行单向数据传输，构成一个单向的光传输系统。光纤通信是利用光纤传递光脉冲来进行数据通信的。通常是一个光脉冲表示比特 1，而无光脉冲则表示比特 0。

5.3.2 光纤的分类

随着光纤的发展，时至今日，光纤种类是千奇斗艳，分类标准也非常多，主要有按光传播模式划分、按光纤芯折射率分布划分和按光纤的套塑层封装形式划分这 3 种。

1. 按光传播模式划分

光在光纤内是以不断反射的方式由一端传递到另一端。我们要明白，并不是任何模式的光波都能在光纤中传输，每种光纤都只允许某些特定模式的光波通过，而其他模式的光波在光纤中无法传播。由于光纤本身的反射率大于光纤外围材料的反射率，所以射入光纤的光线必须以一定的入射角度进入才能在光纤中传递。这个入射角度就称之为一种模式。

按照光信号传播模式划分，把光纤分为两大类：单模光纤（Single-Mode Fiber，SMF）和多模光纤（Multi-Mode Fiber，MMF）。

（1）单模光纤。

在光纤通信中，有些光纤的纤芯较小，如果光信号不是与光纤轴方向平行的角度照射（如图 5-14 所示），可能在经过光纤包层反射后往包层方向有折射，不会把光信号全反射到光纤上继续传播，造成信号丢失，所以在这种光纤上只允许平行照射（与光纤轴方向一致，称之为“基模”）的光波信号通过，而不允许其他角度照射的光信号通过，这种光纤就叫做“单模光纤”。

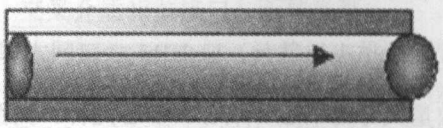


图 5-14 单模光纤光传播模式

单模光纤的纤芯直径一般很小，通常在 8μm～10μm 范围内，光缆外径一般为 125μm（单模光纤中一般也包括了多条纤芯）。单模光纤通常是携带一个波长（通常是 1.31μm 或 1.55μm）的光信号。因为单模光纤完全避免了折射和反射，所以光信号损耗很小，适合于长距离的光纤通信，如目前在有线电视中就是采用单模光纤。

【注意】在单模光纤中也允许多种波长不同的光信号通过，只是这时光信号必须是以与光纤轴方向平行输入。而且单模光纤也不是指单芯光纤，同样可以有多芯。

（2）多模光纤。

相对于单模光纤来说，多模光纤的情形要复杂许多。总体来说，多模光纤是可以在一根光纤上同时传播几种模式的光信号的光纤，如图 5-15 所示。从中可以看出，这些光照射纤芯的角度和从包层反射的角度都不一样。

虽然在多模光纤中是允许光信号以与光纤轴非平行的角度输入，但是又不是任意角度都可以。因为只有以某些特定角度射入纤芯和包层交界面的光波才能相互得到加强，继续传播至远方，否则可能不同光信号最终被抵消或者削弱。也就是说，光波进入纤芯、包层交界面的角度为某些特定值的光波才不会被削弱，而形成一组一组的传输波。

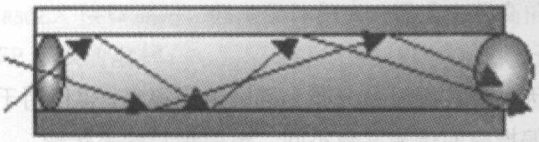


图 5-15 多模光纤传播模式

当多模光纤中同时传输多种模式的光信号时，不同模式光信号的传输方向不是连续改变的，致使它们在光纤中的传输路径也不同：沿光纤轴前进的光走的路径最短，而与轴线交角大的光所走的路径则较长。

多模光纤的纤芯直径较粗，通常为 50μm 或 62.5μm，光缆外径为 125μm。由于存在反射甚至折射，光信号在传输过程中会有损耗，所以多模光纤主要适用于短距离传输，通常在 2km 范围内。

【说明】光纤中传输的光信号按波长可分为以下 3 种：

- 短波：波长在 0.8μm～0.9μm（800nm～900nm）之间，用于多模光纤通信。
- 长波：波长在 1.25μm～1.35μm（1250nm～1350nm）之间，可同时用于单模、多模光纤通信。
- 超长波：波长在 1.53μm～1.58μm（1530nm～1580nm）之间 ，用于单模光纤通信。

2. 按光纤芯折射率分布划分

这是按照光纤中心与四周的折射率变化情况来进行划分的，分为以下几种：

（1）阶跃型光纤（Step Index Fiber，SIF）。

阶跃型光纤又称“突变型光纤”，单模和早期多模光纤多属此类。SIF 光纤纤芯的主要特点是：纤芯的折射率是均匀的，但在纤芯与包层之间的界面上折射率有一不连续的阶跃性突变。纤芯直径为 50μm～100μm，光线以曲折形状传播，信号畸变大。

（2）渐变型光纤（Graded Index Fiber，GIF）。

渐变型光纤又称“梯度光纤”，单模和多模光纤都有此类型。其特点是纤芯中心的折射率最大，沿径向往外逐步变小，形成一个连续渐变的梯度。GIF 光纤的纤芯直径为 50μm，光线以正弦形状传播，信号畸变小。

（3）单模环形光纤（Ring Fiber）。

单模环形光纤折射率分布与突变型光纤相似，仅用于单模光纤中。它的纤芯直径为 8μm～12μm，光线以直线形状沿纤芯中心轴线方向传播，折射率最高，信号畸变小。

（4）单模 W 形光纤（W-Fiber）。

单模 W 形光纤又称为“双包层光纤”，因其折射率分布像 W 形而得名，也仅用于单模光纤中。它的横截面也分为 3 个区域，每个区域内的折射率也都是均匀的，中心的折射率最高，中间区域折射率最低，最外面区域折射率介于两者之间。可以实现 10Gb/s 容量 100km 超大容量超长距离的信号传送。

（5）单模三角形光纤。

单模三角形光纤的纤芯折射率分布曲线为三角形，是一种新型单模光纤，适合于密集波分复用

的长距离系统使用。

（6）单模椭圆形光纤。

单模椭圆形光纤也是因其纤芯折射率分布为椭圆形而得名，也仅用于单模光纤。这种光纤具有双折射特性，即两个正交偏振模的传输常数不同，适用于长距离传输。

3. 按光纤的套塑层封装形式划分

按照光纤的套塑层封装形式可分为“紧套光纤”和“松套光纤”两类。

紧套光纤中光纤被套管紧紧地箍住，不能在其中松动，与塑料套层是一个整体结构。而松套光纤可在套管层中松动，套管内填充油膏，以防水渗入。若一根管内含有多根光纤，则称为松套光纤束。紧套光纤和松套光纤的外观及截面结构如图 5-16 所示。

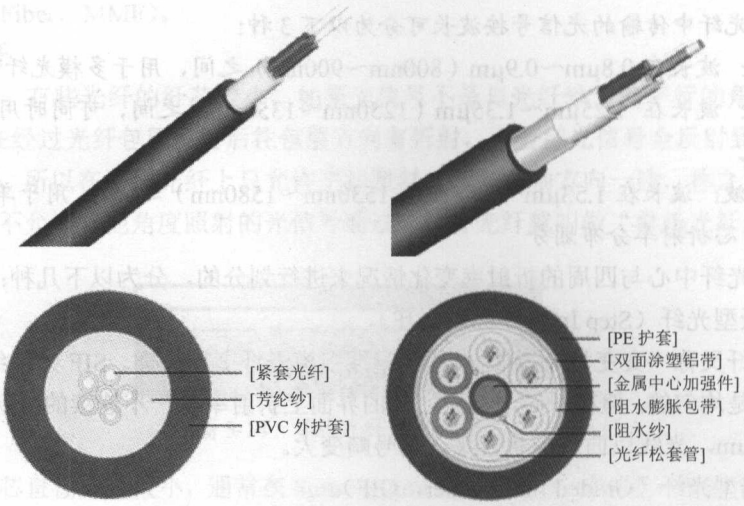


图 5-16　紧套光纤和松套光纤的外观及截面结构

其实光纤的分类非常多，如还可以按一条光纤中所含光芯数量分，目前常见的有 4 芯、6 芯、8 芯、10 芯、12 芯和 14 芯等，当然也有单芯的。注意这是指芯数，而不是指光传输模式中的单模和多模。无论单模还是多模光纤都可以有单芯和多芯光纤。

5.3.3　光纤模块

光纤这种传输介质比较昂贵，所以在网络设备上一般对这种传输介质的支持是采用可选功能模块的方式来提供的，当然也有直接采用固定配置的各种类型光纤端口的。

在光纤模块中我们看到最多的就是 GBIC（Giga Bitrate Interface Converter，千兆位接口转换器）和 SFP（Small Form-factor Pluggable，小型可拔插件），如图 5-17 所示。

GBIC 是将千兆位电信号转换为光信号的接口器件，插入到千兆位以太网端口/插槽内，负责将端口与光纤网络连接在一起，支持热插拔，并在同一个模块上支持 IEEE 802.3z 标准的 1000BaseSX、

1000BaseLX/LH 或 1000BaseZX 接口混用。但由于体积大，很难实现高的端口密度，所以现在基本上已全面被体积更小的 SFP 模块所取代。

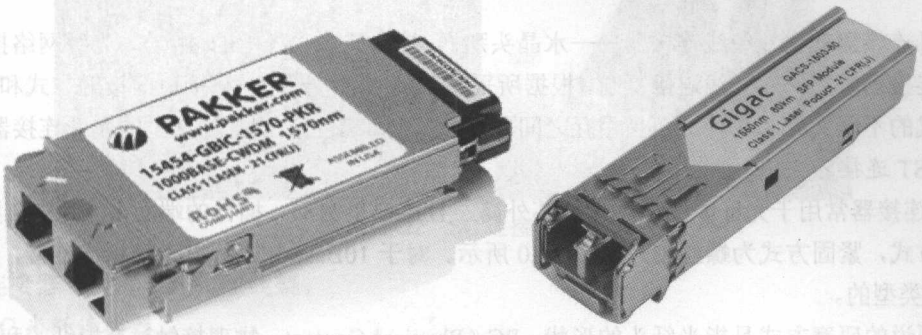


图 5-17　GBIC 和 SFP 光纤模块

SFP 可以简单地理解为 GBIC 的升级版本，体积比 GBIC 模块减小一半，可以在相同的面板上配置多出一倍以上的端口数量，其他功能基本和 GBIC 一致。目前更多是采用 SFP 模块，因为它体积小，同体积下可以扩展更多的端口数。

从图 5-17 可以看出，无论是 GBIC 还是 SFP 模块，都会有两个端口（或者偶数个端口），其原因是光纤默认是不支持全双工模式的，它需要一条光纤用于发送数据，一条光纤用于接收数据。不过，目前高档一些的设备都提供如图 5-18 所示的 SFP 模块，上面只有一个端口，接一条光纤。原因就是它采用了 WDM（波分复用）技术，使得一条光纤可以划分成两个信道，进行全双工传输（模块上那个双向箭头就表示可以双向传输）。

前面介绍的 GBIC 和 SFP 是两个千兆以太网模块，对于现在万兆（10G）以太网来说，有新的光纤模块，那就是 XFP（“X”就代表“十”的意思），如图 5-19 所示。XFP 模块一般都支持 WDM（通常是 DWDM，即密集波分复用）技术，所以通常一个模块就接一条光纤，可以实现全双工传输。

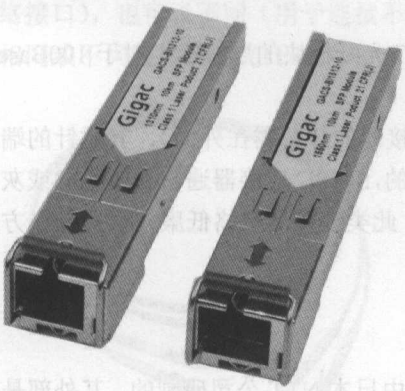


图 5-18　单端口的全双工 SFP 模块

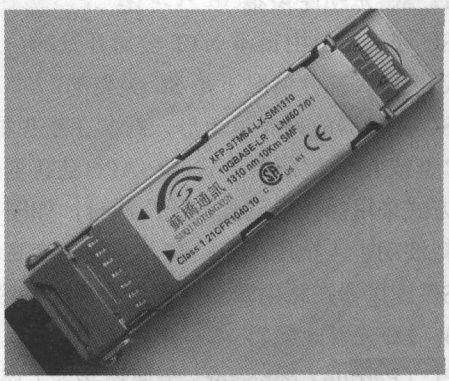


图 5-19　XFP 光纤模块

当然还有许多其他类型网络的光纤模块，而且需要注意，不同品牌设备所用的光纤模块一般都

是私制的，即一般不能通用。

5.3.4 光纤连接器

光纤连接器就像双绞线连接器——水晶头那样，是安装在光纤上的，用于与光纤网络接口连接。但光纤连接器要远比双绞线连接复杂，根据所采用的插入、与光纤网络接口的接触方式和连接器的锁定方式的不同又分为几种，而且相互之间不可以混用。下面介绍几种常见的光纤连接器。

1. ST 连接器

ST 连接器常用于光纤配线架，光纤芯外露，且接口呈圆形，插针的端面多采用 PC 型或 APC 型研磨方式，紧固方式为螺丝扣，如图 5-20 所示。对于 10Base-F 光纤以太网标准连接，连接器通常是 ST 类型的。

前面说的研磨方式是指光纤头的形状，PC（Physical Contact，物理接触）是指纤芯和光纤网络接口的嵌套是平面型的紧密接触，而 APC（Angled Physical Contact，呈一定角度的物理接触）是指纤芯和光纤网络接口的嵌套是有一定角度的紧密接触。

由于 APC 的结构与 PC 完全不同，如果用光纤法兰盘（也就是“光纤适配器”，是光纤活动连接器对的连接部件，图 5-21 列出了主要的 4 种适配器）将这两种连接器连接，就会损坏连接器的光纤端面。

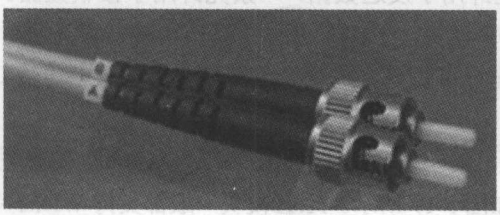


图 5-20 ST 连接器

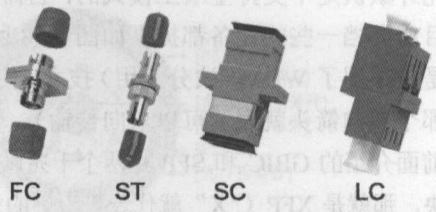


图 5-21 4 种主要的光纤适配器

2. SC 光纤连接器

SC 连接器是由日本 NTT 公司开发的，是连接 GBIC 光纤模块的连接器，对于 100Base-FX 来说，连接器大部分情况下是 SC 类型的。

SC 连接器的外壳呈矩形，纤芯在接头里面（ST 连接器的纤芯露在外面），其插针的端面多采用 PC 型或 APC 型研磨方式(SC-APC 连接器通常是绿色的，SC-PC 连接器通常是蓝色的或灰色的)，紧固方式采用插拔销闩式，无须旋转，如图 5-22 所示。此类连接器价格低廉，插拔操作方便，介入损耗波动小，抗压强度较高，安装密度高。

SC 连接器在路由器和交换机上用得最多。

3. FC 光纤连接器

FC（Ferrule Connector，金属连接器）连接器最早是由日本 NTT 公司研制的，其外部是金属套（连接更紧），紧固方式为螺丝扣，如图 5-23 所示。最早的 FC 连接器采用陶瓷插针的对接端面是平面接触方式，后来采用对接端面呈球面的插针，而外部结构没有改变。

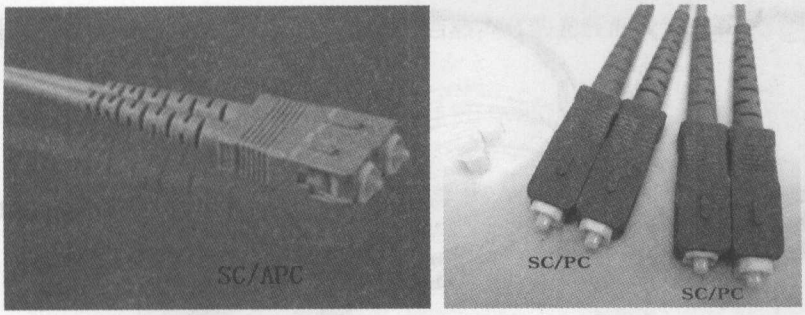


图 5-22 SC 连接器

FC 连接器一般用于连接配线架。

4. LC 光纤连接器

LC 连接器是由著名的 Bell（贝尔）研究所研究开发出来的，是连接 SFP 模块的连接器，采用操作方便的模块化插孔（RJ）闩锁机理制成，如图 5-24 所示。该连接器所采用的插针和套筒的尺寸是普通 SC、FC 类型连接器所用尺寸的一半，提高了光配线架中光纤连接器的密度。

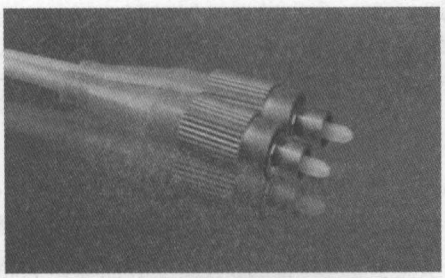


图 5-23 FC 连接器

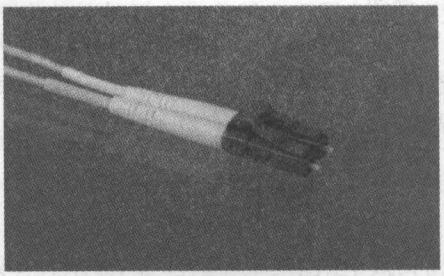


图 5-24 LC 连接器

LC 连接器在路由器上用得比较多。当然，光纤两端的连接器可以一样（用于连接相同类型的光纤网络接口），也可以不同（用于连接不同类型的网络接口），图 5-25 至图 5-29 所示是几种常见的带有相同或不同连接器的光纤。



图 5-25 LC-LC 光纤



图 5-26 SC-SC 光纤

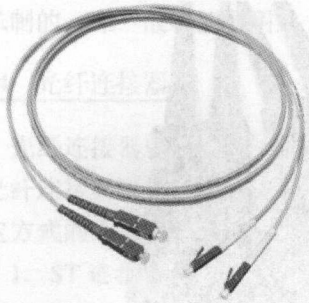


图 5-27　SC-LC 光纤

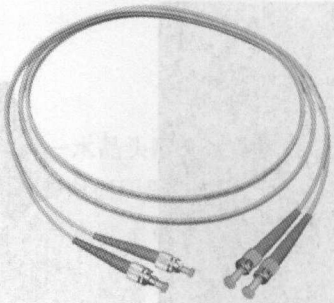


图 5-28　ST-FC 光纤

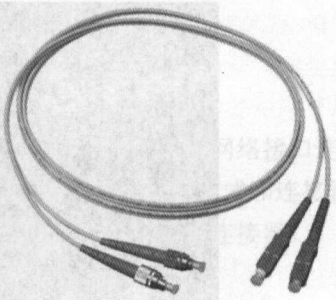


图 5-29　FC-SC 光纤

5.3.5　尾纤与跳纤

在我们做工程时除了要用到整条普通的光纤来进行网络设备连接外，在部署机房时还往往要用到两种特殊用途的短（通常最多只有几米长）光纤——尾纤和跳纤，本节具体介绍这两种光纤及它们的主要用途。

1. 尾纤

“尾纤”（Tail Fiber）又叫做尾线，是只有一端安装了连接器，另一端是一根光缆纤芯（没有安装连接器）的光纤，如图 5-30 所示是一束单根尾纤，如图 5-31 所示是在一条光纤中包括多条尾纤的光纤。

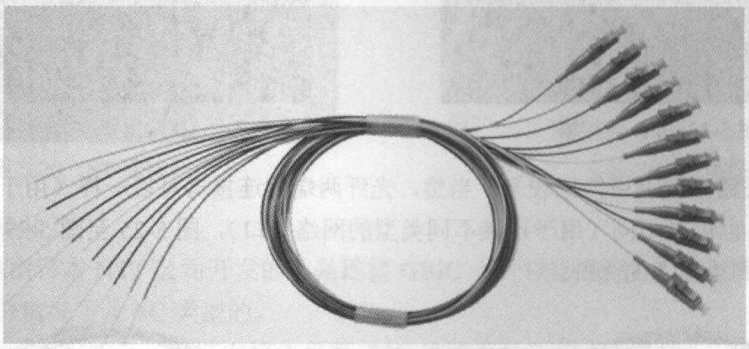


图 5-30　一束单根尾纤

尾纤主要用于与其他光纤进行连接，当然这个连接不是那么简单，必须通过熔接方式（需要通过专门的光纤熔接机进行熔接）把尾纤中没有安装连接器的一端与其他光纤中的纤芯（也没有安装连接器）连接起来。

使用尾纤的主要目的是使光纤另一端的连接器类型更换可以更加灵活，因为通过这种熔接方式可以根据需要随意更换原来光纤中另一端所安装的连接器类型，否则就可能为此要专门购买比较昂贵的整条光纤。如原来的光纤所带的连接器不适合与你所使用的光纤收发器连接，此时只需要把原来光纤中所安装连接器不适用的一端剪掉，露出其纤芯，再与安装了适合连接器的尾纤进行熔接，

这样就不需要四处找两端连接器都匹配的光纤了。当然，有的光纤配线架也要使用尾纤来连接。

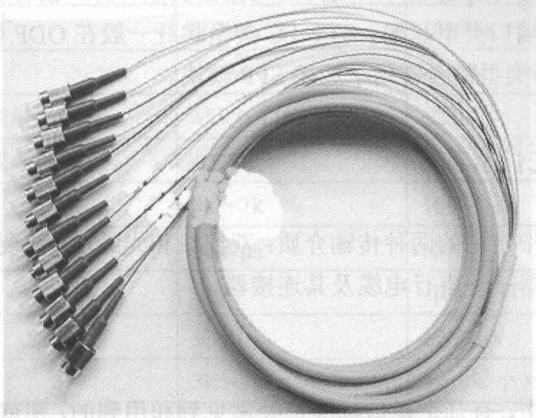


图 5-31 包括多条尾纤的光纤

但这里要说明的是，尾纤也是光纤，所以它同样有单模和多模以及其他类型之分，所以在与光纤进行熔接时光纤类型一定要相同。另外，尾纤一端所安装的连接器也与普通光纤所支持的连接器类型一样，也有 ST、SC、FC、LC 等之分。

2. 跳纤

“跳纤”可以简单地理解为“跳线”，就是用来短距离通过光纤在两台设备间跳线连接的一种光纤。其实它与普通光纤本质上没什么区别，只是它很短，价格便宜，你用普通光纤也可以连接两台光纤设备，但整条光纤通常比较长，价格昂贵。

跳纤与前面介绍的尾纤的相同之处就是一般也是只有几米长，但与尾纤不同的是，跳纤的两端必须同时安装连接器（如图 5-32 所示是两端安装了 FC 连接器的跳纤），而且主要是用于光纤设备之间，或者光纤设备与 ODF（Optical Distribution Frame，光纤配线架）之间，或者 ODF 之间的短距离连接。

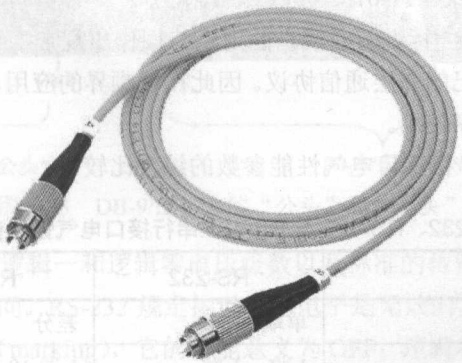


图 5-32 一种类型跳纤

前面说了，跳纤与普通光纤本质是一样的，所以它同样有单模和多模以及其他类型之分。另外，跳纤两端所安装的连接器也与普通光纤所支持的连接器类型一样，也有 ST、SC、FC、LC 等之分，一定要与所连接设备的光端口类型匹配。ST、FC 类型跳纤一般在 ODF 侧采用，SC 类型跳纤一般用于连接 GBIC 模块，LC 类型跳纤常用于连接 SFP 模块。

5.4 串行电缆及连接器

前面介绍了在局域网中常用的两种传输介质：双绞线和光纤，本节要介绍主要在广域网中应用的传输介质及对应的连接器——串行电缆及其连接器。

5.4.1 串行接口标准概述

在计算机广域网通信中，有几个接口标准是经常见到和用到的，那就是一些主要串行接口标准——RS-232、RS-422 和 RS-485，最初都是由电子工业协会（EIA）制定并发布的。

RS-232 接口（又称 EIA RS-232）是目前最常用的一种串行通信接口，但它是非平衡型（即一路信号只在一条数据线上传输）的。它是在 1970 年由美国电子工业协会（EIA）联合贝尔系统、调制解调器厂家及计算机终端生产厂家共同制定的用于串行通信的标准。它的全名是“数据终端设备（DTE）和数据通信设备（DCE）之间串行二进制数据交换接口技术标准”。该标准规定采用一个 25 引脚的 DB-25 连接器，并对连接器每个引脚的功能和电平进行规定。因为 RS-232 接口是非平衡型的，所以它的通信距离短，传输速率也较低。

RS-422 由 RS-232 发展而来，弥补了前面提到的 RS-232 接口通信距离短、速率低的不足。RS-422 标准定义了一种平衡型（一路信号由两根数据线进行反相传输）通信接口，将传输速率提高到 10Mb/s，传输距离延长到 4000 英尺，后来被命名为 TIA/EIA-422A 标准。

为扩展应用范围，EIA 又于 1983 年在 RS-422 基础上制定了 RS-485 标准，增加了多点、双向通信能力，即允许多个发送器连接到同一条总线上，同时增加了发送器的驱动能力和冲突保护特性，扩展了总线共模范围，后命名为 TIA/EIA-485A 标准。

RS-232、RS-422 与 RS-485 标准只对接口的电气特性作出规定，而不涉及接插件、电缆或协议，在此基础上用户可以建立自己的高层通信协议。因此在视频界的应用，许多厂家都建立了一套高层通信协议，或公开或厂家独家使用。

表 5-4 所示是以上 3 种串行接口电气性能参数的综合比较。

表 5-4　RS-232、RS-422 与 RS-485 串行接口电气性能参数综合比较

性能参数	RS-232	RS-422	RS-485
工作方式	单端	差分	差分
节点数	1 收、1 发	1 发 10 收	1 发 32 收
最大传输电缆长度	50 英尺	400 英尺	400 英尺

续表

性能参数	RS-232	RS-422	RS-485
最大传输速率	20Kb/s	10Mb/s	10Mb/s
最大驱动输出电压	±25V	-0.25V～+6V	-7V～+12V
驱动器输出信号电平（负载最小值）	±5V～±15V	±2.0V	±1.5V
驱动器输出信号电平（空载最大值）	±25V	±6V	±6V
驱动器负载阻抗（Ω）	3k～7k	100	54
摆率（最大值）	30V/μs	N/A	N/A
接收器输入电压范围	±15V	-10V～+10V	-7V～+12V
接收器输入门限	±3V	±200mV	±200mV
接收器输入电阻（Ω）	3k～7k	4k（最小）	≥12k
驱动器共模电压		-3V～+3V	-1V～+3V
接收器共模电压		-7V～+7V	-7V～+12V

5.4.2 RS-232 串行接口

目前 RS-232 是 PC 机与通信工业中应用最广泛的一种串行接口，其中 EIA（Electronic Industry Association）代表美国电子工业协会，RS（Ecommeded Standard）代表推荐标准，232 是标识号。RS-232 被定义为一种在低速率串行通信中增加通信距离的单端标准。

一个完整的 RS-232 接口有 22 根线，采用标准的 25 芯插头座（DB-25）。除此之处，目前广泛应用的还有一种 9 芯的 RS-232 接口（DB-9）。它们的外观都是 D 形的。不过，对接的两个接口又分为针式的“公头”和孔式的“母头”两种，DB-9“母头”和“公头”与 DB-25 的“母头”和“公头”如图 5-33 所示。

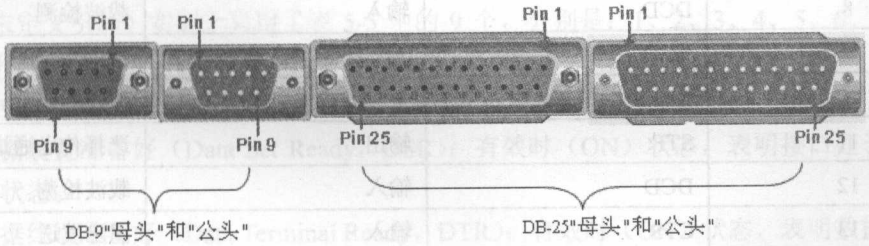


图 5-33 DB-9/DB-25 的“公头”和“母头”

在 RS-232 标准中定义了逻辑一和逻辑零电压级数以及标准的传输速率和连接器类型。信号大小在正的和负的 3V～15V 之间。RS-232 规定接近零的电平是无效的，逻辑一规定为负电平，有效负电平的信号状态称为传号（marking），它的功能意义为 OFF；逻辑零规定为正电平，有效正电平的信号状态称为空号（spacing），它的功能意义为 ON。根据设备供电电源的不同，±5、±10、±12

和±15 这样的电平都是可能的。

1. RS-232 连接器

RS-232 标准中规定的设备可以分为数据终端设备（DTE）和数据通信设备（DCE）两类，这种分类定义了不同的线路用来发送和接收信号。一般来说，计算机和终端设备有 DTE 连接器，调制解调器和打印机有 DCE 连接器。

RS-232 标准中指定了 20 个不同的信号连接，是由 25 个 D-sub（微型 D 类）管脚构成的 DB-25 连接器。但很多设备只是用了其中的一小部分管脚，出于节省资金和空间的考虑不少机器采用较小的连接器，特别是 9 管脚的 D-sub 或 DB-9 型连接器被广泛使用于绝大多数自 IBM 的 AT 机之后的 PC 机和其他许多设备上。

因为 RS-232 到目前为止经历了好多不同的版本，最近的版本号为 E，它相对目前广泛应用的 C 版本来说，电气性能改进了不少，也对连接器中的 25 个管脚进行了充分利用（只有 2 个予以保留）。表 5-5 所示是 25 芯 RS-232E 接口的信号和管脚分配，而表 5-6 所示是在 PC 机、调制解调器和路由器等网络设备中使用的 9 芯 RS-232E 接口的信号和管脚分配。

表 5-5　25 芯 RS-232E 接口的信号和管脚分配

引脚号	信号缩写符	信号方向	说明
1	PE	公共端	连接器外壳
2	TXD	输出	发送数据
3	RXD	输入	接收数据
4	RTS	输出	请示发送
5	CTS	输入	清除发送
6	DSR	输入	数据设备准备好
7	SG	公共端	信号地
8	DCD	输入	载波检测
9	—	—	保留
10	—	—	保留
11	STF	输出	选择传送通道
12	DCD	输入	载波检测
13	CTS	输入	清除发送
14	TXD	输出	发送数据
15	TCK	输入	发送时钟
16	RXD	输入	接收数据
17	RCK	输入	接收器定时
18	LL	输出	本地回路控制

续表

引脚号	信号缩写符	信号方向	说明
19	RTS	输出	请示发送
20	DTR	输出	数据终端准备好
21	RL	输出	远程回路控制
22	RI	输入	振铃指示器
23	DSR	—	数据装置准备好
24	XCK	输出	发送器定时
25	TI	输入	测试指示器

表 5-6　9 芯 RS-232E 接口的信号和管脚分配

引脚号	信号缩写符	信号方向	说明
1	DCD	输入	载波检测
2	RXD	输入	接收数据
3	TXD	输出	发送数据
4	DTR	输出	数据终端准备好
5	GND	公共端	信号地
6	DSR	输入	数据装置准备好
7	RTS	输出	请示发送
8	CTS	输入	清除发送
9	RI	输入	振铃指示

在 RS-232C 版本中，DB-25 的 25 个引脚（4 个数据引脚、11 个控制信号引脚、3 个定时引脚、7 个备用和未定义引脚）实际上只用了表 5-5 中的 9 个，分别是：1、2、3、4、5、6、8、20、22 脚。它们的作用如下：

（1）控制信号引脚（4、5、6、8、20 和 22 号 6 个引脚）。

- 数据装置准备好（Data Set Ready，DSR）：有效时（ON）状态，表明接口处于可以使用的状态。
- 数据终端准备好（Data Terminal Ready，DTR）：有效时（ON）状态，表明数据终端可以使用。

这两个信号有时连到电源上，一上电就立即有效。但这两个设备状态信号有效，只表示设备本身可用，并不说明通信链路可以开始进行通信了，能否开始进行通信要由下面的控制信号决定。

- 请求发送（Request to Send，RTS）：用来表示 DTE 请求 DCE 发送数据，即当终端要发送数据时使该信号有效（ON 状态），向 DTE 设备请求发送。

- 允许发送（Clear to send，CTS）：用来表示 DCE 准备好接收 DTE 发来的数据，是对请求发送信号 RTS 的响应信号。当 Modem 之类的设备已准备好接收终端传来的数据并向前发送时，使该信号有效，通知终端开始沿发送数据线 TxD 发送数据。这对 RTS/CTS 请求应答联络信号用于半双工 Modem 系统中发送方式和接收方式之间的切换；在全双工系统中作发送方式和接收方式之间的切换。在全双工系统中，因配置了双向通道，故不需要 RTS/CTS 联络信号，使其变高。
- 数据载波检出（Data Carrier dectection-DCD）：也叫“接收线信号检出”（Received Line Detection，RLSD），用来表示 DCE 已接通通信链路，告知 DTE 准备接收数据。当本地的 Modem 收到由通信链路另一端（远地）的 Modem 送来的载波信号时，使 RLSD 信号有效，通知终端准备接收，并且由 Modem 将接收下来的载波信号解调成数字数据后，沿接收数据线 RxD 送到终端。
- 振铃指示（Ringing，RI）：当 Modem 收到交换台送来的振铃呼叫信号时，使该信号有效（ON 状态），通知终端，已被呼叫。

（2）数据发送与接收线（2 和 3 号两个引脚）。

- 发送数据（Transmitted data，TxD）：通过 TxD 终端将串行数据发送到 Modem（DTE→DCE）。
- 接收数据（Received data，RxD）：通过 RxD 终端接收从 Modem 发来的串行数据（DCE→DTE）。

（3）地线。

SG（7 脚）、PE（1 脚）分别用来接信号地和保护地信号线，无方向。

上述控制信号线何时有效、何时无效的顺序表示了接口信号的传送过程。例如，只有当 DSR 和 DTR 都处于有效（ON）状态时才能在 DTE 和 DCE 之间进行传送操作。若 DTE 要发送数据，则预先将 DTR 线置成有效（ON）状态，等 CTS 线上收到有效（ON）状态的回答后才能在 TxD 线上发送串行数据。这种顺序的规定对半双工的通信线路特别有用，因为半双工的通信才能确定 DCE 已由接收方向改为发送方向，这时线路才能开始发送。

使用 DB-9 连接器，则可作为提供多功能 I/O 卡或主板上 COM1 和 COM2 两个串行接口的连接器。它只提供异步通信的 9 个信号。DB-25 型连接器的引脚分配与 DB-25 型引脚信号完全不同。因此，若与配接 DB-25 型连接器的 DCE 设备连接，必须使用专门的电缆线。

在连接距离上，如果通信速率低于 20Kb/s 时，RS-232C 所直接连接的最大物理距离为 15 米（50 英尺）。如图 5-34 所示是一条两端分别安装了 DB-9 公、母连接器的 RS-232C 电缆；而图 5-35 所示则是一条一端安装了 DB-9 连接器，另一端安装了 DB-25 连接器的 RS-232C 电缆。

2. 工作原理

信号的定义是从 DTE 设备的角度出发的，TD、DTR 和 RTS 信号是由 DTE 产生的，RD、DSR、CTS、DCD 和 RI 信号是由 DCE 产生的。接地信号是所有连接都公用的，在有的标准中接地信号外部有两个管脚事实上是同一个信号。如果两个通信设备的距离相差得很远或者是有两个不同的供电系统

供电，那么地信号在两个设备间会不一样，从而导致通信失败，跟踪描述这样的情形是很困难的。

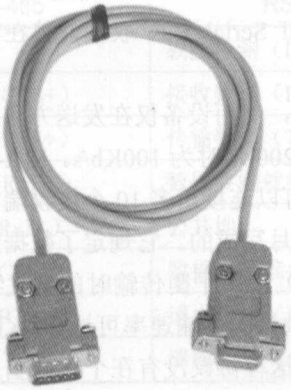


图 5-34 两端均安装 DB-9 连接器的 RS-232C 电缆

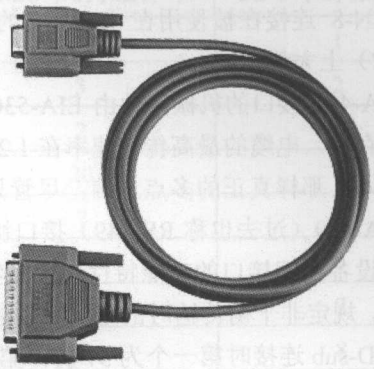


图 5-35 两端分别安装 DB-9 和 DB-25 连接器的 RS-232C 电缆

收、发端的数据信号是相对于信号地的。如从 DTE 设备发出的数据在使用 DB25 连接器时是 2 脚相对 7 脚（信号地）的电平。典型的 RS-232 信号在正负电平之间摆动，在发送数据时，发送端驱动器输出正电平在+5～+15V 之间，负电平在-5～-15V 之间。当无数据传输时，线上为 TTL，从开始传送数据到结束线上电平从 TTL 电平到 RS-232 电平再返回 TTL 电平。接收器典型的工作电平在+3～+12V 与-3～-12V 之间。由于发送电平与接收电平的差仅为 2V～3V 左右，所以其共模抑制能力差，再加上双绞线上的分布电容，其传送距离最大约为 15 米，最高速率为 20Kb/s。RS-232 是为点对点（即只用一对收、发设备）通信而设计的，其驱动器负载为 3kΩ～7kΩ，所以 RS-232 适合本地设备之间的通信。

【注意】RS-232C 标准中所提到的“发送”和“接收”都是站在 DTE 立场上，而不是站在 DCE 的立场来定义的。由于在计算机系统中，往往是 CPU 和 I/O 设备之间传送信息，两者都是 DTE，因此双方都能发送和接收。

5.4.3 其他 EIA 标准接口

除了前面介绍的 EIA-232 接口外，在 EIA 接口标准中，常见的串行接口协议还有诸如 EIA-422、EIA-449、EIA-485、EIA-530 等，本节予以简单介绍。

EIA-422（过去称为 RS-422）是采用 4 线、全双工、差分传输、多点通信的数据传输接口协议。与下面将要介绍的 EIA-485 不同的是 EIA-422 不允许出现多个发送端，而只能有多个接收端（EIA-232 接口也可以有多个接收端）。如 EIA-232 型可以有两端口、四端口和八端口等几种型号，而 EIA-422 型接口可以有四端口、八端口，甚至十端口等几种型号。EIA-422 解决方案传输信号的距离和速度比标准 EIA-232 通信更远、更快，并且一般都能抗电子干扰和电涌。当电缆线的长度为 12 米（40 码）时传输速率可以达到 10Mb/s。由于 EIA-422 解决方案具有上述优点，因此成

为工业和制造应用以及分布广泛的销售经营企业的首选产品。

EIA-422 的通常用途是作为 RS-232 的扩展。一个变种的和 RS-232 兼容的 EIA-422 作为 mini-DIN-8 连接在被使用在 iMac 上的因特尔的 Universal Serial Bus 取代前曾在苹果麦金托什（MAC）上大量使用。

EIA-422 接口的机械特性由 EIA-530 或 EIA-449 规定，然而设备仅在发送方和接收方成对出现时才存在。电缆的最高传输速率在 1.2 米时为 10Mb/s，1200 米时为 100Kb/s。EIA-422 不能实现像 EIA-485 那样真正的多点通信，尽管只有一个发送端就可以连接最多 10 个接收端。

EIA-449（过去也称 RS-449）接口协议是于 1992 年 9 月写成的。它规定了数据终端设备和数据通信设备之间接口的功能特性和机械特性。规定 RS-449 采用平衡传输时的电气特性的协议是 RS-422，规定非平衡传输时的电气特性的协议是 RS-423，数据的传输速率可达 200Kb/s。协议规定了两个 D-sub 连接时第一个为 37 脚，第二个为 9 脚。尽管这种协议没有在个人电脑上使用，这种接口在大型数据交换服务器上还是很常见的。

EIA-485（过去叫做 RS-485 或 RS485）接口协议是电力特性规定为 2 线、全双工、多点通信的标准。它的电力特性和 RS-232 大不一样。用缆线两端的电压差值来表示传递信号。1 端的电压标识为逻辑 1，另一端标识为逻辑 0。两端的电压差最小为 0.2V 以上时有效，任何不大于 12V 或者不小于-7V 的差值对接收端都被认为是正确的。

EIA-485 仅仅规定了接收端和发送端的电力特性，没有规定或推荐任何数据协议。EIA-485 可以应用于配置便宜的广域网和采用单机发送，多机接收通信连接，可提供高速的数据通信速率（10 米时为 35Mb/s，1200 米时为 100Kb/s）。EIA-485 和 EIA-422 一样使用双绞线进行高电压差分平衡传输，可以进行大面积长距离传输（最长为 1200 米）。与 EIA-422 不同的是，EIA-422 采用不可转换的单发送端，EIA-485 的发送端需要设置为发送模式，这使得 EIA-485 可以使用双线模式实现真正的多点双向通信。

EIA-485 推荐使用在点对点网络中，可以是线型、总线型网络，不能是星型、环型网络。理想情况下 EIA-485 需要两个终接电阻，其阻值要求等于传输电缆的特性阻抗。没有特性阻抗的话，当所有的设备都静止或者没有能量的时候就会产生噪声。没有终接电阻的话，会使得较快速的发送端产生多个数据信号的边缘，这其中的一些是不正确的。EIA-485 不能使用于星型或者环型的拓扑结构中，主要是由于在这些结构中 EIA-485 接口中过低或者过高的终接电阻可以产生电磁干扰。

EIA-485 在使用四线时可以和 EIA-422 一样实现全双工，还可以实现真正的多点通信。在某些限制条件下 EIA-485 和 EIA-422 可以实现相互的连接。如 SCSI-2 和 SCSI-3 控制卡通常使用这种标准的设备来作为物理层。EIA-485 接口同样可以在一些工厂的项目控制机器上看到，来实现工厂不同楼层之间的数据通信。它可以抵抗机械设备和焊接设备的电磁干扰。

EIA-485 还可以在大型音频系统中使用，如在音乐厅和剧院中通过这种接口的设备就可以使用普通的计算机来运行一些特殊的软件实现远距离音频设备的控制。EIA-485 通过 XLR 标准的线缆连接的设备大量用于麦克风上，从而实现舞台和控制台之间的连接而不需要预设线路。表 5-7 列出了 EIA-485 的一些特性和引脚分配以及与 RS-232 的比较。

表 5-7 EIA-485 的一些特性和引脚分配以及与 RS-232 的比较

EIA-485	RS-232	DB-25 中的引脚序号	DB-9 中的引脚序号
公共地	载波检测（DCD）	8	1
清除发送+（CTS+）	接收数据（RD）	3	2
准备发送+（RTS+）	传输数据（TD）	2	3
接收数据+（RxD+）	数据终端就绪（DTR）	20	4
接收数据-（RxD-）	公共地	7	5
清除发送-（CTS-）	数据设备准备好（DSR）	6	6
准备发送-（RTS-）	请求发送（RTS）	4	7
传输数据+（TxD+）	清除发送（CTS）	5	8

5.4.4 X.21、X.24、X.36 和 EIA-530 接口规范

除了在前面介绍的 RS-232 物理层接口外，目前在广域网的网络连接中还经常用到其他一些物理接口，其中 X.21、X.24、X.35、X.36 和 EIA-530 这 5 种接口最为常见。

X.21 一开始是 CCITT 在 1976 年以建议的形式向外发布的，它定义了用户计算机的 DTE 如何与数字化的 DCE 交换信号的数字接口标准。X.21 的设计目标之一是减少 RS-232 之类的串行接口中的信号线的数目，采用 15 芯连接器代替原来的 25 芯连接器，而且其中仅定义了 8 条接口线。

X.21 的另外一个设计目标是允许接口在比 EIA RS-232C 更长的距离上进行更高速率的数据传输，其电气特性类似于 EIA RS-422 的平衡接口，支持最大的 DTE-DCE 电缆距离为 300 米。X.21 可以按同步传输的半双工或全双工方式运行，传输速率最大可达 10Mb/s。X.21 接口适用于由数字线路（而不是模拟线路）访问公共数据网（PDN）的地区。欧洲网络大多使用 X.21 接口。

X.21 接口的 15 针连接器中定义了 8 条功能线，分成 4 个工作阶段：空闲、呼叫控制、数据传送、清除。X.21 接口规范如表 5-8 所示。X.24 DTE 接口规范、X.24 DCE 接口规范、X.36 接口规范和 EIA-530 接口规范如表 5-9 至表 5-12 所示。

表 5-8 X.21 接口规范

引脚号	信号缩写符	信号方向	说明
1	PG	公共端	设备地
2	T（A）	输出	发送数据（+）
3	C（A）	输出	控制信号（+）
4	R（A）	输入	接收数据（+）
5	I（A）	输入	指示（+）
6	S（A）	输入	信号元素计时（+）
7	B（A）	输入	字节计时（+）

续表

引脚号	信号缩写符	信号方向	说明
8	SG	公共端	信号地
9	T（B）	输出	发送数据（-）
10	C（B）	输出	控制信号（-）
11	R（B）	输入	接收数据（-）
12	I（B）	输入	指示（-）
13	S（B）	输入	信号元素计时（-）
14	B（B）	输入	字节计时（-）

表 5-9　V.24 DTE 接口规范

引脚号	信号缩写符	信号方向	说明
1	PG	保护接地	公共端
2	TXD	输出	发送数据
3	RXD	输入	接收数据
4	RTS	输出	请示发送
5	CTS	输出	清除发送
6	DSR	输入	数据设备准备好
7	SG	公共端	信号地
8	DCD	输出	载波检测
15	RCLK	输入	接收时钟
17	RCLK	输入	接收时钟
20	DTR	输出	数据终端准备好
24	DTECLK	输出	发送时钟（DTE）

表 5-10　V.24 DCE 接口规范

引脚号	信号缩写符	信号方向	说明
1	PG	保护接地	公共端
2	TXD	输出	发送数据
3	RXD	输入	接收数据
4	RTS	输出	请示发送
5	CTS	输出	清除发送
6	DSR	输入	数据设备准备好
7	SG	公共端	信号地

续表

引脚号	信号缩写符	信号方向	说明
8	DCD	输出	载波检测
15	RCLK	输入	接收时钟
17	RCLK	输入	接收时钟
18	TEST	输出	本地回环
20	DTR	输出	数据终端准备好
21	RLB	输出	远程回环
22	RI	输入	振铃指示器
24	DTECLK	输出	发送时钟（DTE）
25	TI	输入	测试指示器

表 5-11 V.36/RS-449 接口规范

引脚号	信号缩写符	信号方向	说明
管壳	PG	公共端	保护接地
4	TXD+	输出	发送数据
5	TRXC+	输入	发送时钟（DCE）
6	RTD+	输入	接收数据
7	RTS+	输出	请求发送
8	RTXC+	输入	接收时钟（DCE）
9	CTS+	输入	清除发送
10	TEST	输出	本地回环激活
11	DSR+	输入	数据设备就绪
12	DTR+	输出	数据终端就绪
13	DCD+	输入	数据载波检测
14	RLB	输出	远程回环
15	RI	输入	振铃指示器
17	CLK+	输出	发送时钟（DCE）
18	TI	输入	测试指示器
19	GND	公共端	公共端返回
22	TXD-	输出	发送数据
23	TRXC-	输入	发送时钟（DCE）
24	RXD-	输入	接收数据

续表

引脚号	信号缩写符	信号方向	说明
25	RTS-	输出	请求发送
26	RTXC-	输入	接收时钟（DCE）
27	CTS-	输入	清除发送
29	DSR-	输入	数据设备就绪
30	DTR-	输出	数据终端就绪
31	DCD-	输入	数据载波检测
35	CLK-	输出	发送时钟（DTE）

表 5-12　EIA-530 接口规范

引脚号	信号缩写符	信号方向	说明
1	PG	公共端	保护接地
2	TXD+	输出	发送数据
3	RXD+	输入	接收数据
4	RTS+	输出	请求发送
5	CTS+	输入	清除发送
6	DSR+	输入	数据设备就绪
7	SG	公共端	信号地
8	DCD+	输入	数据载波检测
9	RTXC-	输入	接收时钟（DCE）
10	DCD-	输入	数据载波检测
11	CLK-	输出	发送时钟（DTE）
12	TRXC-	输入	发送时钟（DCE）
13	CTS-	输出	清除发送
14	TXD-	输出	发送数据
15	TRXC+	输入	发送时钟（DCE）
16	RXD-	输入	接收数据
17	RTXC+	输入	接收时钟（DCE）
18	TEST	输出	本地回环
19	RTS-	输出	请求发送
20	DTR+	输出	数据终端就绪
21	RLB	输出	远程回环

续表

引脚号	信号缩写符	信号方向	说明
22	DSR-	输入	数据设备就绪
23	DTR-	输出	数据终端就绪
24	CLK+	输出	发送时钟（DTE）
25	TI	输入	测试指示器

5.5 信道多路复用技术

在本章前面已经说到，计算机网络中的数据通信是在物理链路的各个信道中传输的（通常在一条物理链路上会有多条信道）。默认情况下，一条信道只传输一路信号，这就有可能造成信道带宽资源的浪费，如在某路信号传输所需的带宽严重低于信道带宽时。

就像一条大马路上的某条车道，因为该车道在划分时就考虑了大货车通过时对车道宽度的要求，所以车道比较宽。而事实上在这条马路上很少有大货车通过。如果仍按交通规则，一条车道只允许一路汽车通过的话，显然就会造成这条大车道利用率的下降。为了解决这个问题，交警可以将这条大车道再划分成两条小车道，在没有大货车通过时允许两部小汽车在这两条小车道上同时通过，可大大提高这条大车道的通行效率。

对于计算机网络中的数据传输也一样，我们可以把一条高带宽的信道划分成多条小带宽的子信道（注意："子信道"是逻辑意义上的，称为"逻辑信道"），这样就可以在一条信道上同时进行多路低带宽需求的信号传输，可大大提高原来这条信道的利用率。这就是我们这里所讲的"信道复用"（Channel Multiplexing）技术。

这里的"复用"就是"共用"或者"共享"的意思，也就是多路信号共用原来这一条物理传输信道进行数据传输。各种信道复用技术都在发送端有对应的"复用器"，而在接收端又有对应的"分用器"。"分用器"的用途就是把原来在同一信道中传输的各子信道中传输的多路信号分离出来。

根据不同的信号和传输介质类型，可以采用不同的信道复用技术，目前主要有：频分复用（Frequency Division Multiplexing，FDM）、时分复用（Time Division Multiplexing，TDM）、波分复用（Wave Division Multiplexing，WDM）和 CDM（Code Division Multiplexing，码分复用）。

5.5.1 频分复用及其原理

"频分复用"（FDM）是按照信道频带宽度进行的信道复用方式（也是一种调制技术），也就是把一条高带宽的信道按照一定的频带宽度划分成若干个低带宽的子频带（或称"子信道"，如图5-36所示，每个子频带的宽度一般是一样的，但也可以不一样），以实现同一时刻通过不同频率的载波调制技术在信道中传输多路基带信号的目的。如有一个频带宽度（也就是信道带宽）为1MHz～

10MHz 的信道，现在如果要划分成 4 个子信道，则可划分为 1MHz～2.5MHz、2.52MHz～5.0MHz、5.2MHz～7.5MHz、7.52MHz～10.0MHz。

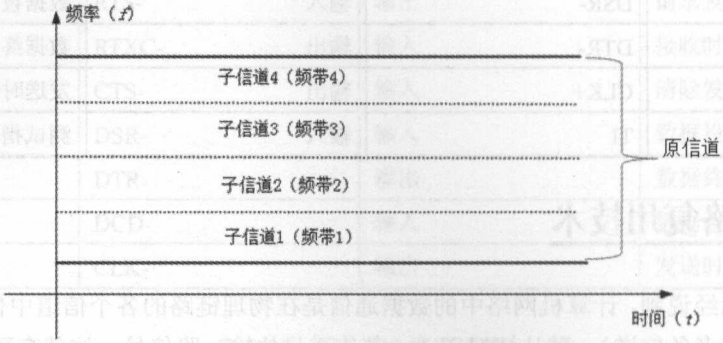


图 5-36　FDM 信道复用原理

1. FDM 基本原理

在 FDM 系统中，信道的可用频带被分成若干个互不交叠的频带，从多条线路输入的多路基带信号通过 FDM 复用器分别用一个不同频带的载波进行调制，然后在同一信道中的不同子信道中传输，但是每路已调制信号只占用原信道中的一部分频带（也就是对应子信道中的频带，如图 5-37 所示），在接收端再通过 FDM 分用器中的带通滤波器将它们分别滤出来，最后通过各线路的解调装置解调接收即可。

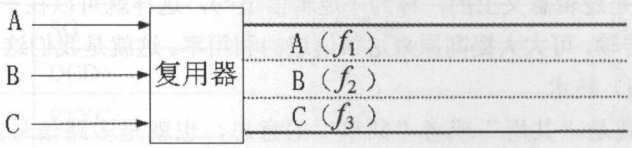


图 5-37　FDM 复用示例

这里就涉及到一个信号工作频带的搬移问题，否则在同一信道中传输的多路信号工作频带可能重叠，自然就达不到信道复用的目的。这个频带搬移工作是由 FDM 复用器来完成的。它可以把进入同一信道的多路信号的工作频带依次搬移到对应子信道的频带中（参见图 5-36），这样多路信号就可以在同一个信道中互不干扰地进行传输。

就像一个合唱中，要使每部分人的声音都能很好地分辨，必须要使一部分人采用低音进行歌唱，一部分人采用中音进行歌唱，还有一部分人采用高音进行歌唱。如果大家都采用同一个音调进行歌唱，就很难区分每部分人唱歌的声音。但在 FDM 中，为了防止相邻子信道中传输信号的相互干扰，不仅各子信道之间的载波频带是完全分隔不重叠的，而且相邻子信道中的频带之间最好要有一定间隙，所以 FDM 对频带的利用率也是相对较低的，当复用的路数不是很多时，可以考虑选择这种信道复用方式。图 5-38 所示是一个 FDM 的应用示例，3 台主机通过一条线路进行数据传输。

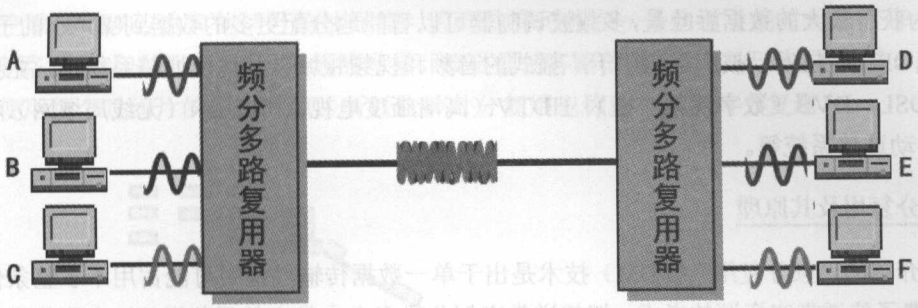


图 5-38 FDM 示例

FDM 系统的优点是信道复用率高、实现简单，是目前模拟通信中经常采用的一种复用方式。但同时 FDM 也存在一些问题，主要表现在各路信号之间存在相互干扰，即串扰。引起串扰的主要原因是滤波器特性不够理想和信道中的非线性特性（就是有突变）造成的已调信号频谱的展宽（也就是一个子信道中的信号频率可能渗透到了另一个子信道中）。调制过程中造成的非线性特性所造成的串扰可以部分地由发送带通滤波器消除，但信道传输中非线性所造成的串扰则无法消除，因而在 FDM 系统中对系统线性的要求很高。另外，在调制过程中合理地选择载波频率并在各路已调信号频谱之间留有一定的保护间隔也是减小串扰的有效措施。

FDM 最典型的应用就是模拟/数字电视、广播系统，电台、电视台使用某个固定特殊的频带或频道来作为某套广播、电视的传输。以前的模拟电话交换系统也是采用 FDM 复用方式的，12 路语音信道被调制到载波上各自占据 4kHz 带宽，可实现在单个物理电路上传输若干条语音信道。不过，现在的数字电话系统基本上已被下节将要介绍的 TDM（时分复用）复用方式所取代了。

2. OFDM

频分复用技术除传统的 FDM 外，还有一种 OFDM（Orthogonal Frequency Division Multiplexing，正交频分复用）。它是一种“多载波调制”（Multi-Carrier Modulation，MCM）技术，可将信道分成若干正交子信道，然后将高速数据信号转换成并行的低速子数据流，调制到在每个子信道上进行传输。

所谓“多载波调制”是指发送端将数据流分解为若干个子数据流，从而使子数据流具有低得多的传输比特速率，再利用这些数据分别去调制若干个载波，然后同时发送出去，采用的是并行化传输技术。与“多载波调制”对应的就是“单载波调制”（Single-Carrier Modulation，SCM），它是指将需要传输的数据流调制到单个载波上进行传送的调制技术。如在第 4 章中介绍的 ASK、FSK、PSK 调制技术都属于单载波调制技术。

全部载波频率有相等的频带（是一个基本振荡频率的整数倍），“正交”是指各个载波的信号频谱是正交（相位相差 90 度）的，也就是各载波的信号频谱可以有交叉，但不会重叠，可以通过某种技术实现完全的分开。由于 OFDM 使用了无干扰的正交多载波调制技术，不仅提高了多路子信道信号的调制效率，而且由于各载波间可以交叉，无需保护频带，这样使得信道中可用频谱的使用效率更高，比 FDM 系统中所要求的信道带宽要小得多。另外，OFDM 技术可动态分配在子信道中

的数据，为获得最大的数据吞吐量，多载波调制器可以智能地分配更多的数据到噪声小的子信道上。

目前 OFDM 技术已被广泛应用于广播式的音频和视频领域以及民用通信系统中，主要的应用包括：ADSL、DVB（数字视频广播）、HDTV（高清晰度电视）、WLAN（无线局域网）和第四代（4G）移动通信系统等。

5.5.2 时分复用及其原理

上节介绍的“频分复用”（FDM）技术是出于单一数据传输过程中可能占用不了整条信道完整带宽，浪费了信道带宽资源的考虑，把信道带宽划分成多个小带宽的子信道，把多路信号的发送分别搬到对应子信道上，以实现信道带宽的更高利用率；而这里的“时分复用”（TDM）则是出于单用户数据传输过程中不可能总有数据传输，通信过程中肯定会存在无数据传输的空闲时段（如用户在输入数据的时候，或者用户在查看数据或网上信息的时候等）的考虑，把整个数据通信过程划分成一个个小的时间段（称为“时间片”），在这些时间片中又划分成多个更小的时间间隙（简称“时隙”），每个时隙可以用于一用户信号的传输，这样就可以使得每一个时间片中一般总会有一个或多个时隙中有信号在里面传输（因为一般不太可能多路信号同时没有数据传输），也可以达到提高信道利用率的目的。

如图 5-39 所示是一个 TDM 示例，其复用器先把整个数据传输过程划分成一个个小的时间片，然后在这些时间片中又划分了 4 个时隙，分别用于传输 3 路信号（相当于 3 个用户，假设为 A、B、C）。

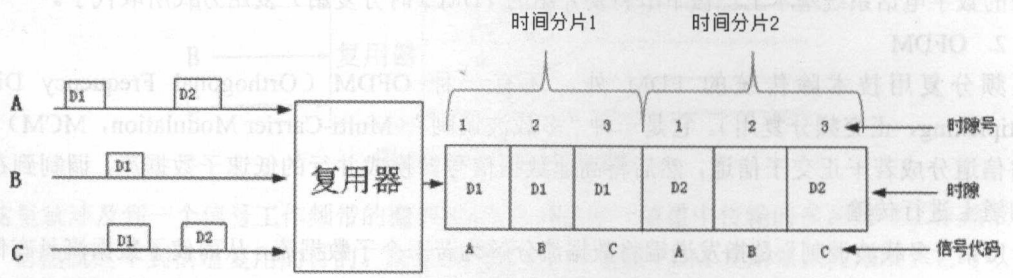


图 5-39 TDM 信道复用示意图

假设在某一时间前，A 和 C 路信号发送了两个数据帧（D1 和 D2），而 B 路信号只发送了一个数据帧 D1，则经过时分复用器复用后，这段时间前发送的 5 个数据帧分别放在两个时间片中，每个时间片划分了可供 3 路信号同时传输的 3 个时隙。此时我们发现在第一个时隙中，A、B、C 三路信号都有数据帧 D1 在传输，而在时间片 2 中，A 和 C 路信号都在它们各自的时隙中有数据 D2 传输，而 B 路信号此时没有数据传输。在接收端通过分复用器即可还原各路信号。

时分多路复用建立在抽样定理基础上，因为抽样定理使连续的基带信号变成在时间上离散的抽样脉冲，这样当抽样脉冲占据较短时间时，在抽样脉冲之间就留出了时间空隙（如图 5-40 所示）。

利用这种空隙便可以传输其他信号的抽样值，因此就有可能在一条信道上同时传送若干个基带信号。时分多路复用以时间作为信号分割的参量，故必须使各路信号在时间轴上互不重叠。由于每路数据总是使用每个时间片的固定时隙，所以这种时分复用也称为“同步时分复用”。

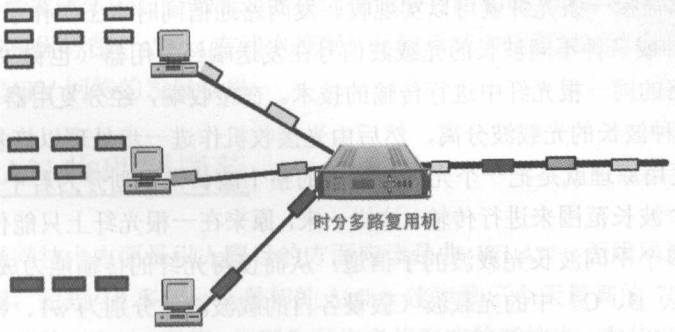


图 5-40 TDM 示例

时分复用的典型例子：PCM（Pulse Code Modulation，脉码调制）信号的传输，把多个话路的 PCM 话音数据用 TDM 的方法装成帧（帧中还包括了帧同步信息和信令信息），每帧在一个时间片内发送，每个时隙承载一路 PCM 信号。

从以上原理的解析可以看出，尽管 TDM 通过划分小的时隙可以提高信道的利用率，但还是会存在一些不彻底的问题。如在图 5-39 中，在第二个时间片中，因 B 路信号此时没有数据发送，所以该时隙仍处于空闲状态，即使其他路信号的传输很忙，也不能占用该时隙进行数据传输。所以才有了一种改进型的 TDM，那就是 STDM（Statistical Time Division Multiplexing，统计时分复用）。

STDM 是一种可动态分配时隙（也就是不为所有用户分配固定的时隙）的 TDM。它可以使用按需分配的技术，即根据用户需求动态分配时隙，以避免传统 TDM 中出现空闲时隙的现象，因为 STDM 复用器中仅为有数据传输的用户分配时隙。同样是因为不为每个用户在每个时间片上分配固定的时隙，所以用户线路的时隙在各时隙中不是周期性出现的，所以 STDM 又称为“异步时分复用”。

与传统 TDM 一样，STDM 也可以与 n 条低速输入用户线路相连。但由于每条输入线路并不是一直有数据输入，因此 STDM 时隙数 k 可小于用户线路数 n。这使得复用信道上的带宽低于各个输入线路带宽之和，或者说对于同样带宽的复用信道，STDM 可以复接更多的用户线路，即提高了信道的利用率。

5.5.3 波分复用及其原理

“波分复用”（WDM）是光信号中的“频分复用”（FDM，针对电信号）技术，又称“光波分复用”，是在单条光纤上同时发送多束不同波长激光的技术。因为光波信号的频率一般非常高，不便于表示，而用波长更便于表示，所以在光信号复用中采用了“波分复用”这个词。其实它与“频

分复用”是一个意思，因为不同波长的光波对应的工作频率也不一样。

试想一下，单独拉一条光纤的成本远比单独拉一条双绞线高，如果在一条光纤中只传输一路信号，肯定是不合算的。再加上，不同光纤的波长并不一样，绝大多数是不能完全占用整条光纤的带宽的。在不采用 WDM 技术的光纤连接中，收、发要用两条光纤，现在新型的光纤连接中通常都支持 WDM，所以只需接一条光纤就可以实现收、发两路通信同时双工工作了。

WDM 是将两种或多种不同波长的光载波信号在发送端经复用器（也称“合波器”）汇合在一起，并耦合到光线路的同一根光纤中进行传输的技术。在接收端，经分复用器（也称“分波器”或“去复用器”）将各种波长的光载波分离，然后由光接收机作进一步处理以恢复原信号。

WDM 的信道复用原理就是把一个光纤信道中的整个波长带宽划分为若干个小波长范围，每路光载波信号占用一个波长范围来进行传输。这样一来，原来在一根光纤上只能传送一个光载波的单一信道变为可传送多个不同波长光载波的子信道，从而使得光纤的传输能力成倍增加。如图 5-41 所示是 3 路光纤（A、B、C）中的光载波（假设各自的载波波长分别为 w1、w2 和 w3）通过合波器后在一条光纤中传输的示例，复用后各路光载波信号的波长带宽分别为 w4、w5 和 w6（对应小于 w1、w2、w3）。

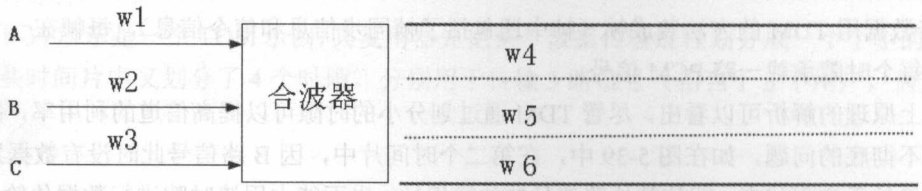


图 5-41　WDM 示意图

光波分复用一般应用波长分割复用器和解复用器（也称合波/分波器）分别置于光纤两端，实现不同光波的耦合与分离。这两个器件的原理是相同的。

光波分复用的技术特点与优势如下：

（1）可灵活增加光纤传输容量。

光波复用技术可充分利用光纤的低损耗波段增加光纤的传输容量，使一根光纤传送信息的物理限度增加一倍至数倍。对已建光纤系统，尤其是早期铺设的芯数不多的光缆，只要原系统有功率余量，则可进一步增容，实现多个单向信号或双向信号的传送而不用对原系统作大改动，具有较强的灵活性。

（2）同时传输多路信号。

光波复用技术使得在同一根光纤中传送两个或多个非同步信号成为可能，而且与数据速率和调制方式无关，在线路中间可以灵活地取出或加入信道。

（3）成本低，维护方便。

由于大量减少了光纤的使用量，大大降低了建设成本。由于光纤数量少，当出现故障时，恢复起来也迅速方便。

（4）可靠性高，应用广泛。

由于系统中有源设备大幅减少，这样就提高了系统的可靠性。目前由于多路载波的光波分复用对光发射机、光接收机等设备要求较高，技术实施有一定难度。同时多纤芯光缆的应用对于传统广播电视传输业务未出现特别紧缺的局面，因而 WDM 的实际应用还不多。但是随着有线电视综合业务的开展，对网络带宽需求的日益增长，各类选择性服务的实施、网络升级改造经济费用的考虑等，光波复用的特点和优势在 CATV（有线电视网）传输系统中逐渐显现出来，表现出广阔的应用前景，甚至将影响 CATV 网络的发展格局。

5.6 主要 WLAN 物理层规范

近年来，在局域网技术方面最引人瞩目的方面应该是非 WLAN（无线局域网）技术莫属。可以说是一年一个跨越，短短 10 多年，从最初的 3Mb/s 速率到了今天最高的 7Gb/s。在其中不仅诞生了各种不同接入速率的 WLAN 规范，也诞生了许多相对应的新技术。本节仅就 WLAN 中的一些典型物理层接入规范及相关主要技术进行简单介绍，有关有线以太网规范和 WLAN 体系架构都将在第 7 章中介绍。

5.6.1 WLAN 物理层规范概述

WLAN 技术的发展真可用“目不暇接”来形容，呈现大步跨越式的发展。在写本书第一版的 2012 年，第一次真正实现百兆 WLAN 接入的 IEEE 802.11n 还刚刚走向市场，可在四年之后的今天，以太网中千兆这个门槛就直接跨过，直奔挑战 10G 以太网、两个最高接入速率可达 7G 的新的 WLAN 规范——IEEE 802.11ac 和 IEEE 802.11ad 相继诞生，并且已进入实质应用。随着这些新的 WLAN 物理层接入规范的出现，也出现了与之对应的 WLAN 数据链路层安全规范。

就像以太网中有许多以太网接入规范在 IEEE 802.3 系列一样，WLAN 中的这些接入规范也集中在一个大的系列即 IEEE 802.11 系列中，也是由 IEEE 802 标准委员会制定的。1990 年 IEEE 802 标准化委员会成立 IEEE 802.11 WLAN 标准工作组，最初的无线局域网标准是 IEEE 802.11 于 1997 年正式发布的，该标准定义物理层和介质访问控制（MAC）规范。物理层定义了数据传输的信号特征和调制，工作在 2.4000～2.4835GHz 频段。这一最初的无线局域网标准主要用于难于布线的环境或移动环境中计算机的无线接入，由于传输速率最高只能达到 2Mb/s，所以业务主要被用于数据的存取。但随着无线局域网应用的不断深入，人们越来越认识到，2Mb/s 的连接速率远远不能满足实际应用需求，于是 IEEE 802 标准委员会不断开发了一系列更高接入速率的新 WLAN 规范。

自从 WLAN 这项技术真正进入实质应用开始，这 10 多年来出现的主要 WLAN 接入规范（按发布时间先后排列）主要有 IEEE 802.11b、IEEE 802.11a、IEEE 802.11g、IEEE 802.11n、IEEE 802.11ad 和 IEEE 802.11ac，它们的基本特性如表 5-13 所示，通过这个表不仅可以了解各 WLAN 接入规范的基本特性，还可以从中了解整个 WLAN 接入规范的发展历程。

表 5-13　主要 WLAN 接入规范及基本特性

特性 / 规范	802.11b	802.11a	802.11g	802.11n	802.11ac	802.11ad
发布时间	1999 年 9 月	1999 年 9 月	2003 年 6 月	2009 年 9 月	2013 年 6 月	2012 年 11 月
工作频段	2.4GHz	5GHz	2.4GHz	2.4GHz/5GHz	5GHz	60GHz
最高接入速率	11Mb/s	54Mb/s	54Mb/s	600Mb/s	7Gb/s	7Gb/s
频带宽度	20MHz	20MHz	20MHz	20MHz/40MHz	40MHz/80MHz/160MHz	2160MHz
调制方式	DSSS 等	OFDM	DSSS/OFDM	OFDM	OFDM	OFDM
兼容性	—	—	802.11b	802.11a/b/g	802.11a/g/n	802.11a/g/n/ac

WLAN 从最初的 3Mb/s 到目前最高的接入速率可达 7Gb/s，已接近以太网目前在企业网络中所应用的最高速率——10Gb/s，而且仅用了不到 20 年时间，这在当初是想都不敢想的。而且据说 IEEE 已在开发下一代，直接到达 10Gb/s 接入速率的 IEEE 802.11ax 接入规范，预计 2019 年可以正式发布。下面对表 5-13 中各 WLAN 接入规范的主要特性进行介绍。

5.6.2　IEEE 802.11b 规范主要特性

在 WLAN 的发展历史中，真正具有实用无线连接的 WLAN 标准还是 1999 年 9 月正式发布的 IEEE 802.11b。但由于接入速率太低，早已被全面淘汰了。该规范的主要特性包括：

（1）工作频段。

IEEE 802.11b 规范工作在免费的 2.4GHz 频段，室内有效传输距离为 35 米，室外有效传输距离为 140 米。

（2）传输速率。

IEEE 802.11b 规范的最高传输速率为 11Mb/s，还可以根据实际网络环境调整为 1Mb/s、2Mb/s、5.5Mb/s。

（3）调制方法。

IEEE 802.11b 规范可根据不同接入速率采用不同的调制技术：传输速率为 1Mb/s 和 2Mb/s 时，采用原来 IEEE 802.11 规范中的 DSSS（Direct Sequence Spread Spectrum，直接序列扩展）、DBPSK（Differential Binary Phase Shift Keying，差分二相位键控）、DQPSK（Differential Quadrature Phase Shift Keying，差分四相位键控）数字调制方法；传输速率为 5.5Mb/s 和 11Mb/s 时，采用 CCK（Complementary Code Keying，互补编码键控）调制方法。

【说明】DSSS 是先将信号源与一定的 PN（Pseudo Noise，伪噪声）码进行混合，然后通过 DBPSK、DQPSK 相位键控技术将原来信号中的“0”或“1”比特用 11 个 chips 的巴克序列（Barker sequence）：{+1,-1,+1,+1,-1,+1,+1,+1,-1,-1,-1}进行替代，扩展频谱，使得原来较高功率、较窄频带的频率变成具有较宽频带的低功率频率。简单来说，DSSS 利用高频率的信号，通过各个调变技术

进行扩展频，将发送端的频谱信号频带展宽，而在接收端用相同的展频技术去进行译码，把展频后的信号还原成原始的信息。DSSS 具有抗干扰能力强、强的抗多径干扰能力、对其他电台干扰小、抗截获能力强、可以同频工作、便于实现多址通信等优点。

（4）信道划分。

IEEE 802.11b 规范全球使用的是同一无线电模型，共有 11 个信道，每个信道带宽为 22MHz，但相邻信道间只有 5MHz 带宽不重叠（也就是会重叠 17MHz 带宽），如图 5-42 所示。

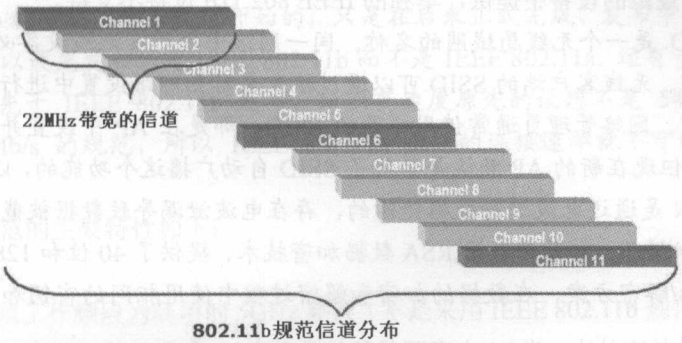


图 5-42　IEEE 802.11b 信道分布

因为每两个相邻信道都会有大部分的频段重叠，所以在 IEEE 802.11b 规范的整个频段中，真正完全不重叠的信道最多只有 3 个，如图 5-43 所示。

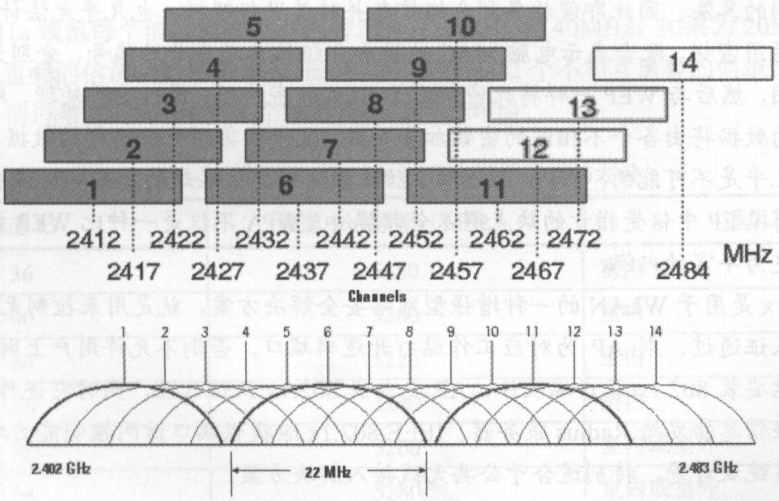


图 5-43　IEEE 802.11b 规范信道划分及频率覆盖范围

这些不重叠的信道有 5 组，即：1、6 和 11 号信道，2 和 7 号信道，3 和 8 号信道，4 和 9 号信道，5 和 10 号信道，每个区域 WLAN 网络只能选择一组，且推荐使用 1、6 和 11 这组信道，这样

就可以在同一个区域 WLAN 网络中部署 3 台 AP 了。为保证这部分区域所使用的信号信道不能互相覆盖，各 AP 要使用不同的信道，最好选择完全不重叠的一组信道。

（5）主要安全技术。

IEEE 802.11b 规范主要采用的安全技术包括：SSID（Service Set Identifier，服务集标识符）和 WEP（Wired Equivalent Privacy，有线等效保密）链路加密。但在 2003 年后生产的 IEEE 802.11b 规范的 WLAN 一般还支持 WPA 和 IEEE 80.1x 安全技术，但这通常是在同时支持 IEEE 802.11b 和 IEEE 802.11g 两种规范的设备中提供，单独的 IEEE 802.11b 设备不支持。

【说明】SSID 是一个无线局域网的名称。同一网络中的所有无线设备必须具有相同的 SSID 才能实现相互通信。无线客户端的 SSID 可以通过输入名称到网络设置中进行手动设置，或者自动设置为不指定或空。网络管理员通常使用公开的 SSID，那是在 AP 中设置并广播给所有在网络范围内的无线设备。但现在新的 AP 默认是禁用了 SSID 自动广播这个功能的，以提高网络的安全性。另外，因为 WLAN 是通过电波进行数据传输的，存在电波泄漏导致数据被截听的风险，所以采用 WEP 链路加密保护措施。它是源自于 RSA 数据加密技术，提供了 40 位和 128 位长度的密钥机制，但它是一个对称加/解密方案，在数据的加密和解密过程中使用相同的密钥和算法，存在较大的安全风险。

WPA（Wi-Fi Protected Access，Wi-Fi 受保护访问）作为 IEEE 802.11i 标准的子集，包含了认证、加密和数据完整性校验 3 个组成部分，是一个完整的安全方案，其核心是 IEEE 802.1x 和 TKIP（Temporal Key Integrity Protocol，临时密钥完整性协议）。

WPA 是一种继承了 WEP 基本原理而又解决了 WEP 缺点的新技术，使用 128 位钥。由于加强了生成加密密钥的算法，因此即便收集到分组信息并对其进行解析，也几乎无法计算出通用密钥。其原理为根据通用密钥，配合表示电脑 MAC 地址和分组信息顺序号的编号，分别为每个分组信息生成不同的密钥，然后与 WEP 一样将此密钥用于 RC4 加密处理。通过这种处理，所有客户端的分组信息所交换的数据将由各个不相同的密钥加密而成。无论收集到多少这样的数据，要想破解出原始的通用密钥几乎是不可能的。WPA 还追加了防止数据中途被篡改的功能和认证功能。由于具备这些功能，此前 WEP 中倍受指责的缺点得以全部解决。WPA 不仅是一种比 WEP 更为强大的加密方法，而且有更为丰富的内涵。

IEEE 802.1x 是用于 WLAN 的一种增强型网络安全解决方案，就是用来控制无线工作站对 AP 的访问。如果认证通过，则 AP 为对应工作站打开逻辑端口，否则不允许用户上网。IEEE 802.1x 要求无线工作站安装 802.1x 客户端软件，AP 要内嵌 802.1x 认证代理，同时它还作为 Radius 客户端将用户的认证信息转发给 Radius 服务器。IEEE 802.1x 除提供端口访问控制能力之外，还提供基于用户的认证系统及计费，特别适合于公共无线接入解决方案。

5.6.3 IEEE 802.11a 规范主要特性

虽然 IEEE 802.11b 规范的 11Mb/s 传输速率比起最初 IEEE 802.11 的 3Mb/s 来说有了几倍的提高，但这也只是理论数值，在实际应用环境中的有效速率还不到理论值的一半。为了继续提高传输

速率，IEEE 802 工作小组继续了下一个规范的开发，那就是 2001 年底发布的 IEEE 802.11a。目前事实上也几乎全面被淘汰了。



在这里要说明的一件事就是，为什么最先推出的规范命名为 IEEE 802.11b，而后来推出的规范反而是 IEEE 802.11a。那是因为，这两个规范是分属于两个不同的小组。事实上 IEEE 802.11a 规范与 IEEE 802.11b 的研制工作是同时开始的，只是在后来正式完成、发布中，IEEE 802.11b 规范却走在了前面，所以最先发布的是 IEEE 802.11b 而不是 IEEE 802.11a。还有一点，那就是 IEEE 802.11a 规范本来要先于 IEEE 802.11b 发布，所以其速度原先的设想不是 54Mb/s，只是 IEEE 802.11b 发布了 11Mb/s 的规范，所以 IEEE 802.11a 规范的连接速率就不可能再低于或者接近 11Mb/s，只能超过。

IEEE 802.11a 规范的主要特性如下：

（1）工作频段。

IEEE 802.11a 规范工作频段为商用的 5GHz 频段（不是采用 IEEE 802.11b 规范中的 2.4GHz 免费频段，所以不与 IEEE 802.11b 设备兼容），室内有效传输距离为 35 米，室外有效传输距离为 120 米。

（2）传输速率。

IEEE 802.11a 规范的最高数据传输速率为 54Mb/s，根据实际网络环境，还可以调整为 6Mb/s、9Mb/s、12Mb/s、18Mb/s、36Mb/s、48Mb/s。

（3）信道划分。

IEEE 802.11a 规范每个信道的带宽有两种选择：20MHz 和 40MHz，如果为 20MHz 带宽，则共有 24 个不相互重叠的信道；如果是 40MHz 带宽，则共有 12 个不相互重叠的信道。表 5-14 列出了当采用 20MHz 信道带宽时可用的 24 个信道及所适用的环境。

表 5-14　IEEE 802.11a 规范中可用信道的划分

信道 ID	信道中心频率（MHz）	适用环境
36	5180	室内
40	5200	室内
44	5220	室内
48	5240	室内
52	5260	室内或室外
56	5280	室内或室外
60	5300	室内或室外
64	5320	室内或室外
100	5500	室内或室外

续表

信道 ID	信道中心频率（MHz）	适用环境
104	5520	室内或室外
108	5540	室内或室外
112	5560	室内或室外
116	5580	室内或室外
120	5600	室内或室外
124	5620	室内或室外
128	5640	室内或室外
132	5660	室内或室外
136	5680	室内或室外
140	5700	室内或室外
149	5745	主要用于室外
153	5765	主要用于室外
157	5785	主要用于室外
161	5805	主要用于室外
165	5825	主要用于室外

（4）调制方法。

IEEE 802.11a 规范采用 OFDM（Orthogonal Frequency Division Multiplexing，正交频分复用）调制扩频技术，可提高信道的利用率。在 IEEE 802.11a 规范的 OFDM 调制技术中，20MHz 的信道带宽情况下，将整个频带分成了 64 个子载波，实际使用的是其中的 52 个（其余频带是保护用途的频带间隔），其中 48 个是用于传输数据的数据子载波，另外 4 个是先导子载波（Pilot Carrier，就是载波里面没有携带任何数据），每个子载波的间距为 0.3125MHz。另外，IEEE 802.11a 可以采用 BPSK（二相移相键控）、QPSK（四相移相键控）、16-QAM（Quadrature Amplitude Modulation，正交振幅调制）或 64-QAM 调制技术。

OFDM 技术是将信道分成若干正交子信道，将高速数据信号转换成并行的低速子数据流，调制到每个子信道上进行传输。正交信号可以通过在接收端采用相关技术来分开，这样可以减少子信道之间的相互干扰。

（5）主要安全技术。

IEEE 802.11a 规范在安全方面一开始也主要使用 WEP 加密技术和 SSID。但在 2003 年后生产的 IEEE 802.11a 规范的 WLAN 一般还支持 WPA 和 IEEE 80.1x 安全技术，但这通常是在同时支持 IEEE 802.11a 和 IEEE 802.11g 两种规范的设备中提供，单独的 IEEE 802.11a 设备不支持。

5.6.4 IEEE 802.11g 规范主要特性

虽然 IEEE 802.11a 规范的速度已经非常高了，但由于 IEEE 802.11b 和 IEEE 802.11a 两个规范的工作频段不一样，相互不兼容，致使一些原先购买 IEEE 802.11b 规范的无线网络设备在新的 802.11a 网络中不能用，于是推出一个能兼容 IEEE 802.11b 规范同时提供更高接入速率的新规范，那就是 2003 年 6 月 IEEE 正式推出 IEEE 802.11g 规范。

IEEE 802.11g 规范的主要特性如下：

（1）工作频段。

IEEE 802.11g 规范与 IEEE 802.11b 规范一样工作在免费的 2.4GHz 频段（与 IEEE 802.11b 兼容，但不与 IEEE 802.11a 兼容），室内有效传输距离为 38 米，室外有效传输距离为 140 米。

（2）传输速率。

IEEE 802.11g 规范与 IEEE 802.11a 规范一样的传输速率（54Mb/s），总带宽为 20MHz，如果需要，传输速率可降为 48Mb/s、36Mb/s、24Mb/s、18Mb/s、12Mb/s、9Mb/s 或 6Mb/s。

（3）信道划分。

IEEE 802.11g 规范共划分了 13 个信道，每个信道所覆盖的频率范围参见图 5-43，不同的只是 IEEE 802.11g 规范中 12、13 号这两个信道是可用的。在这 13 个信道中，真正完全不重叠的信道最多也只有 3 个，有 5 种组合，即：1、6 和 11 或 13 号信道，2、7 和 12 号信道，3、8 和 13 号信道，4 和 9 号信道，5 和 10 号信道。每个区域 WLAN 网络只能选择其中一组，美国是推荐使用 1、6 和 11 这组信道，但世界上大多数国家还推荐使用 1、5、9、13 这组信道，这样就可以在同一个区域 WLAN 网络中部署 4 台 AP 了。为保证这部分区域所使用的信号信道不能互相覆盖，各 AP 要使用不同的信道，最好选择完全不重叠的一组信道。

（4）调制方法。

IEEE 802.11g 规范同时采用了 IEEE 802.11a 中的 OFDM 和 IEEE 802.11b 中的 DSSS、CCK 等多种调制技术。

（5）主要安全技术。

在安全性方面，IEEE 802.11g 规范全面支持 IEEE 802.11i 标准中的 WPA、WPA2、EAP（Extensible Authentication，可扩展身份认证）AES（Advanced Encryption Standard，高级加密标准）加密，以及用于访问控制的 IEEE 802.1x 标准。

【说明】在以上这 3 个主要无线局域网接入规范之外，我们还可以见到诸如 IEEE 802.11b+、IEEE 802.11a+和 IEEE 802.11g+这 3 个所谓对应规范的增强版，它们的传输速度也相应增强，达到原有规范的 2 倍，分别为 22Mb/s、108Mb/s 和 108Mb/s。但这 3 个所谓的增强版规范并不是正式的规范，而是一些无线网络设备开发商自己制定的企业规范，它们的兼容性较差，通常只能与本企业的某些无线网络设备相兼容。

5.6.5 IEEE 802.11n 规范主要特性

为了实现高带宽、高质量的 WLAN 服务，使无线局域网达到以太网的性能水平，802.11 任务组 n（TGn）应运而生。IEEE 802.11n 是 2009 年 9 月正式发布的 IEEE 新的 802.11 规范，也是目前最主要应用的 WLAN 接入规范。IEEE 802.11n 是在 IEEE 802.11g 和 IEEE 802.11a 之上发展起来的一项技术，最大的特点是速率提升，理论速率最高可达 600Mb/s（目前业界主流为 300Mb/s），可以工作在 2.4GHz 和 5GHz 两个频段。其主要特性如下：

（1）工作频段。

IEEE 802.11n 规范可以工作在 2.4GHz 和 5GHz 两个频段，所以它可以全面向下兼容以前发布的 IEEE 802.11b/a/g 这 3 个规范。

（2）覆盖范围。

IEEE 802.11n 采用智能天线技术，通过多组独立天线组成的天线阵列可以动态调整波束，保证让 WLAN 用户接收到稳定的信号，并可以减少其他信号的干扰。因此其覆盖范围可以扩大到好几平方公里，使 WLAN 的移动性极大提高。

（3）传输速率。

IEEE 802.11n 的接入速率相较以前版本有大幅提升，这主要得益于它将 MIMO（多进多出）与 OFDM（正交频分复用）技术相结合而应用的 MIMO OFDM 技术，提高了无线传输质量。IEEE 802.11n 规范在标准带宽（20MHz）单倍 MIMO 上支持的速率有：7.2Mb/s、14.4Mb/s、21.7Mb/s、28.9Mb/s、43.3Mb/s、57.8Mb/s、65Mb/s、72.2Mb/s，使用标准带宽和 4 倍 MIMO 时，最高速率为 300Mb/s；在双倍带宽（40MHz）和 4×4MIMO（即同时收、发 4 路数据流）时，最高速率可达 600Mb/s，是 IEEE 802.11g 的 10 倍多。

（4）信道划分。

IEEE 802.11n 规范总共可以有 15 个不相互重叠的信道，其中在 2.4GHz 频段中有 3 个不相互重叠的信道（每个信道 22MHz 带宽），在 5GHz 频段中有 12 个不相互重叠的信道（每个信道 40MHz 带宽）。另外，通过将两个相邻的 20MHz 带宽捆绑在一起组成一个 40MHz 通信带宽，在实际工作时可以作为两个 20MHz 带宽的信道使用（一个为主带宽，一个为次带宽，收发数据时既可以 40MHz 的带宽工作，也可以单个 20MHz 的带宽工作），这样可将速率提高一倍。同时，对于 IEEE 802.11a/b/g，为了防止相邻信道干扰，20MHz 带宽的信道在其两侧预留了一小部分的带宽边界。而通过频带绑定技术，这些预留的带宽也可以用来通信，从而进一步提高了吞吐量。

（5）调制方法。

IEEE 802.11n 规范采用了 IEEE 802.11g 规范中相同的 OFDM 调制技术，只是选择的正交载波数更多。OFDM 是可将信道分成许多进行窄带调制和传输的正交子信道，并使每个子信道上的信号带宽小于信道的相关带宽，用以减少各个载波之间的相互干扰，同时提高频谱利用率的技术。MIMO（Multiple-Input and Multiple-Output，多进多出）与 OFDM 技术的结合就产生了 MIMO OFDM 技术，它通过在 OFDM 传输系统中采用阵列天线实现空间分集，提高了信号质量，并增加了多径

的容限，使无线网络的有效传输速率有质的提升。

（6）主要安全技术。

IEEE 802.11n 规范与 IEEE 802.11g 规范所使用的主要安全技术类似，主要就是 IEEE 802.11i 所引入的 WPA、WPA2 和 AES 加密以及 IEEE 802.1x 访问控制技术。

5.6.6 IEEE 802.11ac 规范主要特性

虽然 IEEE 802.11n 同时支持 2.4GHz 和 5.0GHz 两个频段，但目前大多数设备仍采用的是公用 2.4GHz 频段设计。而 2.4GHz 频段中本身的可用信道较少，同时还有其他工作于 2.4GHz 频段的设备（如蓝牙、微波炉、无线监视摄像机等）的干扰，使得即使在双空间流 40Mhz 频宽下其连接速率只能达到 300Mb/s，是其理论值的一半，用户体验并不是很好。IEEE 802.11ac 规范是专门为 5GHz 频段设计的，特有的新射频特点能够将现有的无线局域网的性能吞吐提高到可以与有线千兆级网络相媲美的程度。

从其规范名称上就可以看出，IEEE 802.11ac 规范是基于以前的 IEEE 802.11a 规范开发的，继续工作在 5.0GHz 频段（被称为新 5G 的 WiFi，我们目前主要使用的 WiFi 技术是工作在 2.4GHz 频段的）上以保证向下兼容性，但同时又继承了前一版本——IEEE 802.11n 规范的频带扩充和 MIMO（多进多出）扩容特性，最终使得 IEEE 802.11ac 规范的接入得到极大提高。最基本的接入速率就可以达到 433Mb/s，第一个版本最终达到了 3.2Gb/s，其第二版（Wave 2）最高可以达到近 7Gb/s，使 WLAN 第一次真正进入千兆时代。

IEEE 802.11ac 规范的主要特性如下：

（1）工作频段。

IEEE 802.11ac 采用 IEEE 802.11a 规范的 5.0MHz 频段，所以它可以与以前的 IEEE 802.11a/g/n 版本兼容，保护了用户在 WLAN 设备上的投资。

（2）传输速率。

虽然 IEEE 802.11ac 规范的核心技术主要基于 IEEE 802.11a，但数据传输信道带宽得到了极大的扩充，从原来的 20MHz 扩展到了 40MHz 或 80MHz，甚至有可能达到 160MHz。再加上它继承了 IEEE 802.11n 的 MIMO 技术，所以 IEEE 802.11ac 的接入速率有了相当大幅度的提升。最初的 IEEE 802.11ac 版本（可称为 1.0 版）的基本接入速率可达到 400Mb/s，采用 80MHz 信道宽度的 IEEE 802.11ac 在 3×3 MIMO 模式下就能提供 1.3Gb/s 的下载速度；在 IEEE 802.11ac Wave2 版本（可称为 2.0 版本）下，对原来 IEEE 802.11n 规范中的 MIMO 技术作了改进，称为 MU-MIMO（Multi-User Multiple-Input Multiple-Output，多用户—多进多出）技术。这时如果信道宽度为 160MHz，3×3 MIMO 模式下就可使总的传输速度达到 2.6Gb/s，8×8 MIMO 模式下就可以达到近 7Gb/s 的接入速率。

（3）信道划分。

采用 5.0GHz 频段的频率带宽通常是 5.170GHz～5.835GHz，不同国家或地区对 IEEE 802.11ac 规范使用 5.GHz 频段的监管和要求不一样。以美国为例，如果采用 20MHz 信道带宽，则最多可有 24 个不重叠信道；如果采用 40MHz 信道带宽，则最多可有 11 个不重叠信道；如果采用 80MHz 信

道带宽，则最多可有 5 个不重叠信道；如果采用 160MHz 信道带宽，则不重叠信道最多 2 个。

在我国，目前 IEEE 802.11ac 规范中的 5.0GHz 频段只能使用 5.735GHz～5.835GHz 这个小的频带，而且不能使用 160MHz 这样带宽的信道，对于 20MHz、40MHz 和 80MHz 这 3 种情况，不重叠信道数分别为 5、2、1。

（4）调制方法。

IEEE 802.11ac 使用与 IEEE 802.11a/n 相同的 OFDM 调制技术，在新增了 256-QAM 调制方式（应用于 80MHz 和 160MHz 信道带宽时）的同时能兼容支持 BPSK、QPSK、16-QAM 和 64-QAM 调制方式。256-QAM 的好处在于能提供比 64-QAM 更高 33%的传输速率，这也使得 IEEE 802.11ac 的传输速率相较 IEEE 802.11n 有较大提高。

在采用 20MHz 信道带宽时，与 IEEE 802.11a 一样，也是使用了 64 个子载波中的 52 个，其中 48 个也是用于传输数据的数据子载波，另外 4 个是先导载波，每个信道的带宽为 0.3125MHz；而在采用 40MHz 信道带宽时，一共有 128 个 OFDM 子载波，其中数据子载波为 108 个，先导子载波为 6 个，其余 14 个子载波的频带也是保护用途，每个信道的带宽也为 0.3125MHz。在采用 80MHz 信道带宽时，一共有 256 个 OFDM 子载波，其中数据子载波为 234 个，先导子载波为 8 个，其余 14 个子载波的频带也是保护用途，每个信道的带宽也为 0.3125MHz；在采用 160MHz 信道带宽时，一共有 512 个 OFDM 子载波，其中数据子载波为 468 个，先导子载波为 16 个，其余 28 个子载波的频带也是保护用途，每个信道的带宽也为 0.3125MHz。

（5）安全技术。

IEEE 802.11ac 规范向后兼容 IEEE 802.11 系列中主流的 WLAN 规范，包括 IEEE 802.11i、IEEE 802.11s 和 IEEE 802.11u 等，使得采用 IEEE 802.11ac 的无线 Wi-Fi 能够在安全性方面达到企业级用户的需求。

5.6.7 IEEE 802.11ad 规范主要特性

如果前面介绍的 IEEE 802.11ac 主要是用来接替 IEEE 802.11a/g 规范，主要应用于企业 WLAN 中的话，那同时研发的千兆级 IEEE 802.11ad 规范则主要面向家庭娱乐音/视频设备，且工作在与前面所有 IEEE 802.11 接入规范工作频段不同的 60GHz 频段上。IEEE 802.11ad 规范的主要目标能够使高清视频和无损音频成为可能，为家庭多媒体、智能家居应用带来更完备的高清视频解决方案。

IEEE 802.11ad 规范的出现针对的是多路高清视频和无损音频超过 1Gb/s 传输速率的要求，将被用于实现家庭内部无线高清音视频信号的传输，为家庭多媒体应用带来更完备的高清视频解决方案，你将会看到多千兆吞吐量以及在开放式办公室和封闭房间设置内惊人的传输速率和覆盖范围性能（其传输速率比第二代蓝牙技术快 1000 倍以上）。

IEEE 802.11ad 规范的主要特性如下：

（1）工作频段。

IEEE 802.11ad 区别于以前所有 WLAN 接入规范，使用了独特的高频载波 60GHz 频谱成为它的工作频段，因为在多个国家和地区，这个频率都有大量的空闲，而且可免费使用。以前的接入规

范都是 2.4GHz 或 5.0GHz 频段。但需要注意，工作频率越高，穿墙能力越弱，所以 IEEE 802.11ad 的穿墙能力很弱，基本上可以说仅适用于同一个房间使用。

（2）传输速率。

IEEE 802.11ad 的接入速率与 IEEE 802.11ac wave2 版本规范一样，也可达到 7Gb/s。但是 IEEE 802.11ac 规范是通过 8x8 MIMO、256-QAM 调制技术和 4 个 40MHz 信道的绑定来实现的，IEEE 802.11ad 规范通过 10x10 MIMO 技术，仅需要一个信道（2160MHz 带宽），采用 64-QAM 调制技术即可实现。这样一来，如果有多个传输信道，其传输速率还将呈几倍地增长。此外，IEEE 802.11ad 还在容量、功耗和延迟方面有着 IEEE 11ac 无法比拟的优势，特别是在延迟方面，其延迟通常仅有 10μs，就像有线一样。

（3）信道划分。

采用 60GHz 频段的 IEEE 802.11ad 将整个频段划分成 4 条信道，每个信道带宽为 2.16GHz，这样的信道带宽明显比以前的 IEEE 802.11ac 要宽了，所以它更容易实现单信道的高速率传输，更适用于大容量的视频数据传输。

（4）调制方法。

IEEE 802.11ad 所用的调制技术与 IEEE 802.11ac 类似，既有传统的 OFDM 调制方式，也有一种称为 SC-OFDM（单载波 OFDM）的调制方式，另外在 IEEE 802.11ad 规范中每个信道的频带宽度与 IEEE 802.11ac 相比宽了许多倍。

在 OFDM 调制方式中一共有 512 个子载波，其中数据子载波数为 336 个，先导子载波数为 16 个，另外还有 3 个 DC（直流，相当于无信号电流的“空”状态）子载波，其余频段是保护频段。在 OFDM 调制方式中，如采用 64-QAM 调制模式，则单信道的最高传输速率可达 7Gb/s，且适用于较大型设备（如路由器）上较长距离的传输。而在 SC-OFDM 调制方式中，仅使用一个载波，支持的最大传输速率为 2.5Gb/s，适用于在小型、低功率设备（如手机）上短距离使用。

（5）安全技术。

IEEE 802.11ad 规范建立在 IEEE 802.11 系列标准定义的强大安全机制之上，采用基于硬件的 AES 加密技术。其他相关的安全技术正在开发之中，所以目前基于 IEEE 802.11ad 规范的实质应用还极少见到。

5.7 课后自测题

为了巩固本章所学的内容，请认真做下面的自测题，参考答案见附录。

5.7.1 填空题

1．物理层的主要任务是定义了与传输介质、连接器及其接口相关的_____、_____、_____和_____4 个方面的特性，其中_____又分为_____、_____和_____三大类。

2．传输介质总体上可以分为_____和_____两大类，同轴电缆、光纤属于_____，而

电磁波属于_____。

3．单段双绞线的最大传输距离为_____，单段细同轴电缆的最大传输距离为_____，单段粗同轴电缆的最大传输距离为_____，单段光纤的最大传输距离至少可在_____以上。在计算机网络中使用的同轴电缆的阻抗为_____。

4．5 类双绞线的工作频率为_____，传输带宽为_____，6 类双绞线的工作频率为_____，传输带宽为_____，7 类线的工作频率最低可达_____，传输带宽最高可达_____。

5．按光纤折射率分布的不同主要分为_____、_____、_____、_____、_____和_____6 种，按光纤传输模式的不同可以分为_____和_____两类。

6．GBIC 和 SFP 是两个千兆以太网模块，10Gb/s 的模块称为_____，通常支持_____，所以通常一个模块就接一条光纤，可以实现全双工传输。

7．IEEE 802.11b 规范工作在_____频段，最高传输速率为_____，整个频段中共有_____信道，有以下 5 组信道是没有相互重叠的：_____、_____、_____、_____和_____，但每个区域 WLAN 网络只能选择一组。

8．IEEE 802.11a 规范工作在_____频段，最高传输速率为_____，每个信道的带宽有两种选择：20MHz 和 40MHz，如果为 20MHz 带宽，则共有_____个不相互重叠的信道；如果是 40MHz 带宽，则共有_____个不相互重叠的信道。

9．IEEE 802.11g 规范可同时工作在_____和_____两个频段，最高传输速率为_____，共划分了_____个信道，有以下 5 组信道是没有相互重叠的：_____、_____、_____、_____和_____，但每个区域 WLAN 网络也只能选择一组。

10．IEEE 802.11ac 规范的核心技术主要基于 IEEE 802.11a，继续工作在_____频段，第一版本的最高传输速率可达_____，Wave 2 版本的最高传输速率可接近_____。IEEE 802.11ad 规范工作在_____频段上，单一信道传输速率最高可接近_____。

11．在 MIMO 支持倍数方面，IEEE 802.11n 最多能支持_____，而 IEEE 802.11ac 则是支持_____，而 802.11ad 更是可以支持_____。

5.7.2 选择题（可多选）

1．RJ-45 双绞线以太网接口的电气特性属于（　　）。

A．平衡型　　B．非平衡型　　C．新的非平衡型

2．以下关于传输信道的说法正确的是（　　）。

A．一条传输介质代表一个传输信道　　B．一个传输信道代表一条传输介质

C．一条传输介质上可能有多个传输信道　　D．无线网络没有传输信道

3．下列设备中不属于通信子网的是（　　）。

A．PC 机　　B．网卡　　C．传输介质

D．打印机　　E．交换机

4. 下面（　）传输介质没有非屏蔽类型。

A. 5 类双绞线　　B. 超 5 类双绞线

C. 6 类双绞线　　D. 7 类双绞线

5. 下面（　）传输介质的工作频率可在 200MHz 或以上。

A. 5 类双绞线　　B. 超 5 类双绞线

C. 6 类双绞线　　D. 7 类双绞线

6. 下列说法正确的是（　）。

A. 一条光纤中只能有一个信道

B. 一条光纤中只能有一束光通过

C. 一条光纤中可以通过多个波段的光波

D. 一条光纤只能通过一个波段的光波

7. 单模光纤通常所使用的光信号波长为（　）。

A. 1.31μm　　B. 1.55μm　　C. 10μm

D. 62.5μm　　E. 50μm

8. 下列光纤连接器中主要用于配线架的是（　）。

A. ST　　B. SC　　C. FC　　D. LC

9. 下列关于直通线和交叉线的说法不正确的是（　）。

A. 直通线的两端水晶头的线序是一样的，交叉线的两端水晶头的线序不一样

B. 交叉线一端的 1、3 号引脚分别与对端的 2、6 号引脚对应

C. 直通线连接的是相同类型设备，交叉线连接的是不同类型设备

D. 交叉线的一端采用 EIA/TIA 568B 标准，另一端采用 EIA/TIA 568A 标准

10. 下列关于 FDM 的说法正确的是（　）。

A. FDM 是基于信道的频带来划分多个子信道的，同时适用于双绞线和光纤网络

B. FDM 划分出的每个子信道所使用的载波频率都一样

C. 在 FDM 中，不同支路的数据会采用经过调制后的子信道进行传输

D. FDM 的最主要优点是划分后的各子信道间不会存在相互串扰问题

11. 下列关于 TDM 和 STDM 的说法不正确的是（　）。

A. TDM 是把一个传输时间单位划分成多个时隙，每个时隙固定用来传输一支路数据

B. 在 TDM 中，如果某支路当前无数据发送时，则其对应时隙可用来传输其他支路数据

C. TDM 的典型应用是 PCM 调制

D. STDM 又称异步 TDM，因为它在某支路无数据传输时不分配时隙

12. 下列关于 WDM 的说法正确的是（　）。

A. WDM 是基于光信号频率进行搬移的一种复用技术，所以也称为光信号中的 FDM

B. 在 WDM 中，不同频率的光信号会搬移到不同的子信道中进行传输

C. WDM 是把光信道中的整个波长带宽划分为多个小波长范围，每个波长范围通过不同

的光载波信号传输一路光信号

D．可通过 WDM 技术实现单光纤的全双工通信

13．在 IEEE 802.11a 规范的 OFDM 调制中，如果采用 20MHz 信道带宽，则其共有的总子载波数、数据子载波数、先导子载波数分别是（　　）。

A．52　38　4　　B．64　52　12　　C．64　48　4　　D．52　48　4

14．下列 WLAN 规范中可工作在 2.4GHz 频带的是（　　）。

A．802.11a　　B．802.11b　　C．802.11g

D．802.11n　　E．802.11ac

15．在 IEEE 802.11n 规范中，要使接入速率达到 600Mb/s，则其使用的信道带宽和 MIMO 倍数分别是（　　）。

A．20MHz　4×4　　B．40MHz　4×4

C．40MHz　3×3　　D．20MHz　8×8

16．下列 WLAN 规范主要应用于家庭娱乐、智能家居的是（　　）。

A．802.11n　　B．802.11g　　C．802.11ac　　D．802.11ad

17．IEEE 802.11ad 只需要采用（　　）调制模式即可实现单信道 7Gb/s 的传输速率。

A．16-QAM　　B．64-QAM　　C．128-QAM　　D．256-QAM

18．IEEE 802.11ad 所采用的 MIMO 倍数是（　　）。

A．3×3　　B．4×4　　C．8×8　　D．10×10

6 数据链路层

要在一条通信线路上传送数据，除了必须建立一条物理链路（“物理层”的功能）外，还必须有一些规程或协议来控制这些数据的传输，以保证被传输数据的正确性。实现这些规程或协议的硬件和软件加到物理链路上，这样就构成了本章将要介绍的“数据链路层”（Data Link Layer，DLL）。设计“数据链路层”的主要目的就是在原始的、有差错的物理传输线路的基础上，采取差错检测、差错控制与流量控制等方法，将有差错的物理线路改进成逻辑上无差错的数据链路，以便向它的上一层——“网络层”提供高质量的服务。就像我们修好了路，还得制定一些交通法规，使路上行驶的车辆必须按照一定的规则行驶，否则可能会经常出现交通事故。

本章是针对广域网中的“数据链路层”和局域网体系结构中的“逻辑链路控制”（LLC）子层的功能及相关技术进行的全面、深入的介绍，非常重要。其中要着重掌握的是数据链路层的链路管理、数据帧封装与解封装、帧组装与同步、差错控制、流量控制这几项功能的实现原理，BSC、SDLC、HDLC 和 PPP 这几种典型的数据链路层协议，二层交换原理。有关局域网体系结构“数据链路层”中的“媒介访问控制”（MAC）子层的功能和相关技术将在下一章中介绍。

本章重点与难点：

- 划分数据链路层的必要性
- 帧封装与解封装原理
- 帧组装与帧同步原理
- 奇偶校验法、CRC、反馈检测、空闲重发请求、连续重发请求等差错控制技术原理
- 海明码的检/纠错原理以及海明码的计算
- 滑动窗口流量控制原理
- HDLC、PPP 帧结构

- PPP 链路的建立、拆除流程，以及 PAP、CHAP 认证原理
- 冲突域和广播域
- 二层交换原理

6.1 数据链路层基础

在第 2 章介绍的所有计算机网络体系结构中都直接或间接地包含了“数据链路层”，它和下面的“物理层”其实本质作用都是一样的，就是用来构建进行网络通信、访问的通道，只不过“物理层”所构建的是一条物理通道，而“数据链路层”构建的是真正用于数据传输的逻辑通道。正因为如此，在目前 Internet 中广泛使用的“TCP/IP 协议体系结构”中，“物理层”和“数据链路层”是集中划分在“网络接口层”这一层之中的。

6.1.1 划分“数据链路层”的必要性

虽然说“物理层”和“数据链路层”的本质作用都是用来构建网络通信、访问的通道，但它们所建立的通信通道是不一样的。但需要说明的一点是，在“物理层”上构建的是“物理链路”，在“数据链路层”上构建的是“逻辑链路”或者“数据链路”，它们是不同的概念，但确实有许多读者朋友分不清楚。

“物理链路”是指在物理层设备（包括传输介质、物理接口和收发器等）和相应物理层通信规程作用下形成的物理线路，是永久存在的且不可删除的（除非物理拆除）；“逻辑链路”则是在通信双方在需要进行数据通信时在数据链路层设备和相应的通信规程作用下建立的逻辑链路，可以是永远存在的（如局域网中的以太网链路），也可以不是永久存在的（如广域网中的链路），要视具体的数据链路层服务类型而定。这里还有一个“链路”的概念，它是指相邻节点之间的那段数据线路。

在看到“物理链路”和“逻辑链路”之间区别的同时，又要看到它们之间的联系，那就是“逻辑链路”必须建立在“物理链路”之上。如果通信双方的物理线路都不通，是不可能建立用于数据传输的逻辑链路的。我们可以这样来理解它们之间的关系，“物理链路”是基础线路，相当于一条公路的路基，而“逻辑链路”是在“物理链路”之上的高级线路，可以理解为在公路上铺设了柏油或者水泥的车道。它们之间的关系如图 6-1 所示。

无论在多么复杂的网络中，从逻辑意义上来讲，真正的数据传输通道就是“数据链路层”中所定义的“数据链路”，只不过在要经过多个网络的数据通信中“数据链路”是分段的，每个网络都有一段链路（在同一个 IP 网段中由二层交换机连接的各段链路是直接相通的），这些链路段连续起来就是整个数据通信的数据链路。如图 6-2 所示是一个有 3 个网段，经过两个路由器（路由器 A 和路由器 B）相连的网络，现假设 PC 用户 A 要向 PC 用户 B 发送一个数据，它的实际数据传输过程如图中各网络物理层之间的实线箭头所示，而逻辑上可以等同各部分数据链路层之间构建的“逻

辑链路”之间的数据转发，如图中虚线箭头所示。

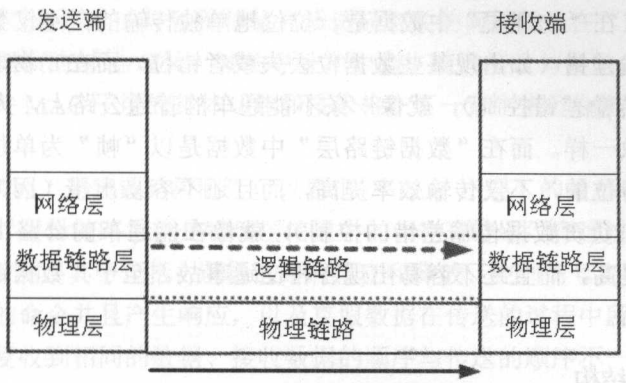


图 6-1 “物理链路”和“逻辑链路”的关系

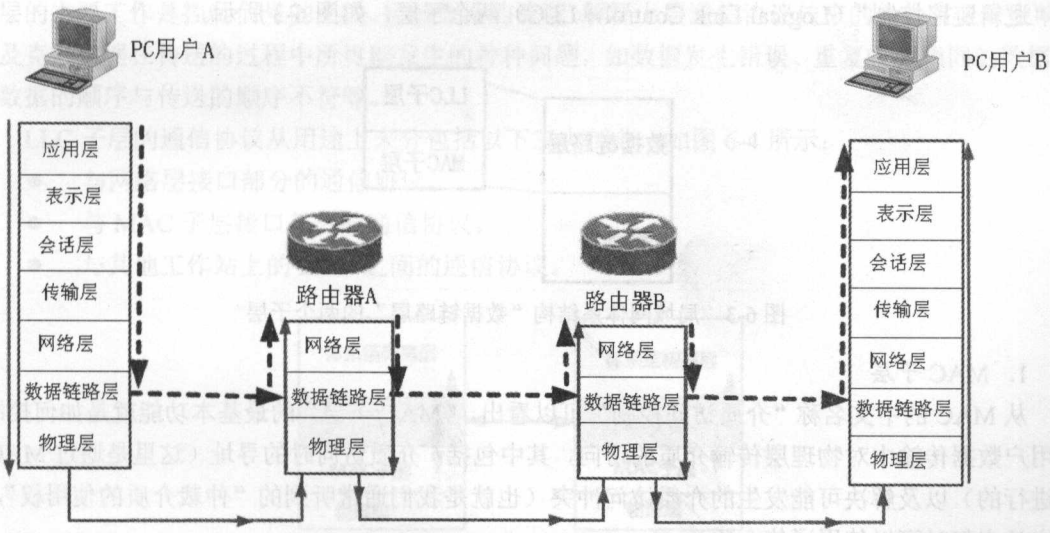


图 6-2 数据在网络中的传输方向

说到这里，可能有的读者朋友立马会问，既然在“物理层”中构建了数据传输通道（也就是物理层中的“信道”），那为什么还要多加一个“数据链路层”的功能呢？其实这里有两方面的主要原因。一是由于物理层传输介质的多样性，通信规程也各不相同，性能不稳定，而“数据链路层”中构建的“逻辑链路”是不考虑不同物理链路上传输介质及其通信规程上的区别的，只是从逻辑意义上构建一条性能稳定、不受传输介质类型影响的逻辑数据传输通道。就像修建公路时，所用的材质也可能不一样，有的是用普通的泥巴，有的是用沙石，还有的是用石材。如果仅靠这些基础材料修建起来的公路，可能通车性能就有很大区别了，有的甚至根本不能通车，只能步行。但如果我们在

这些公路上再统一铺一层钢筋混泥土，铺上水泥，这样这些由不同材料修建的公路就可能满足基本相同的通车性能要求了。

另一个原因是，在“物理层”中数据是一位位地单独传输的，不仅数据传输效率低下，而且容易出现数据传输差错（如出现某些数据位丢失或者错位，而在“物理层”中又没有相应的通信规程进行数据传输差错控制），就像一条不能通车的普通公路上，人只能一个个地步行，还可能出现迷路现象一样。而在“数据链路层”中数据是以“帧”为单位进行传输的，一个帧通常是有数千个比特位的，不仅传输效率提高，而且还不容易出错（因为在“数据链路层”中有专门的通信规程来负责数据传输差错的控制），就像在能通车的公路上以车为单位运载人一样，不仅传输效率提高，而且还不容易出现各种交通事故。至于“数据链路层”的主要功能将在本章后有具体介绍。

6.1.2 数据链路层的结构

在 IEEE 802 系列标准中将数据链路层分为“介质访问控制”（Medium Access Control，MAC）和“逻辑链路控制”（Logical Link Control，LLC）两个子层，如图 6-3 所示。

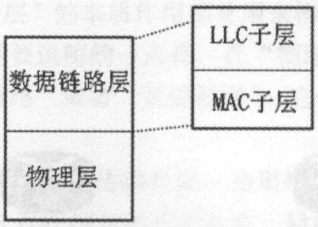


图 6-3　局域网体系结构“数据链路层”的两个子层

1. MAC 子层

从 MAC 的中文名称“介质访问控制”可以看出，“MAC 子层”的最基本功能就是如何控制不同用户数据传输中对物理层传输介质的访问，其中包括：介质访问时的寻址（这里是通过 MAC 地址进行的）以及解决可能发生的介质访问冲突（也就是我们通常听到的“仲裁介质的使用权”，即规定站点何时可以使用通信介质）。

MAC 子层是制定如何使用传输介质的通信协议，如：

- CSMA/CD 通信协议（IEEE 802.3）：规范如何在总线（Bus）型网络结构下使用传输介质。
- Token-Bus 通信协议（IEEE 802.4）：规范如何在总线型网络结构下利用令牌（Token）来控制传输介质的使用。
- Token-Ring 通信协议（IEEE 802.5）：规范如何在令牌环（Ring）网络结构下利用令牌来控制传输介质的使用。
- DQDB 通信协议（IEEE 802.6）：规范如何在双总线（Dual Bus）网络结构下达到以分布式队列（Distributed Queue）的方式来使用传输介质。

- 无线网络协议（IEEE 802.11）：规范如何在无线局域网络（WLAN）结构下控制传输介质的使用。

具体而言，“数据链路层”中与各种传输介质访问有关的问题都放在“MAC 子层”来解决。其主要功能包括：数据帧的封装/卸装、帧的寻址和识别、帧的接收与发送、帧的差错控制、介质访问冲突控制等。有关 MAC 子层的具体功能和技术将在下一章中进行介绍。

2. LLC 子层

从 LLC 的中文名称“逻辑链路控制”可以看出，“LLC 子层”的最基本功能就是负责数据链路层中“逻辑链路”（“逻辑链路”就是物理层信道中的“物理链路”在通过 LLC 子层协议作用后形成的虚拟链路）的控制，其中包括：逻辑链路的建立和释放、控制信号交换、数据流量控制、解释上层通信协议传来的命令并且产生响应，以及克服数据在传送的过程中所可能发生的种种问题，如数据发生错误、重复收到相同的数据、接收数据的顺序与传送的顺序不一致等。

在 LLC 子层方面，IEEE 802 系列标准中只制定了一种标准——IEEE 802.2，各种不同局域网都使用相同的 LLC 子层通信标准。这使得更高层的通信协议不必依赖局域网络的实际架构。LLC 子层的主要工作是控制信号的交换、控制数据的流量、解释上层通信协议传来的命令并且产生响应，以及克服数据在传送的过程中所可能发生的种种问题，如数据发生错误、重复收到相同的数据、接收数据的顺序与传送的顺序不符等。

LLC 子层的通信协议从用途上来分包括以下三大部分，如图 6-4 所示：

- 与网络层接口部分的通信协议。
- 与 MAC 子层接口部分的通信协议。
- 与其他工作站上的 LLC 之间的通信协议。

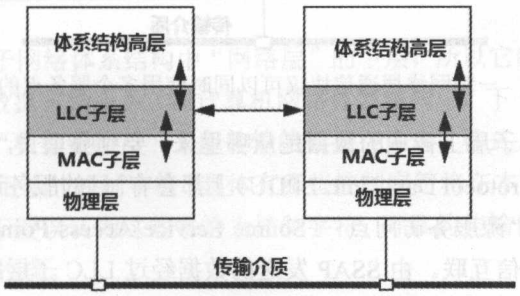


图 6-4 LLC 子层通信协议的三方面应用

由于网络层上可能有许多种通信协议同时存在，而且每一种通信协议又可能同时与多个对象沟通，因此当 LLC 子层从 MAC 子层收到一个数据包时必须能够判断要送给网络层的那一个通信协议。为了达到这种功能，在 LLC 子层中提供了“数据链路层”的 SAP（服务访问点）作为与“网络层”通信交互的接口（每路通信需要一个 SAP 接口，如图 6-5 所示）。SAP 的使用可以简化数据传输的处理过程。如图 6-6 所示，每种网络层的通信协议都可以使用多个 SAP 与其他工作站上的网络层通信协议沟通。

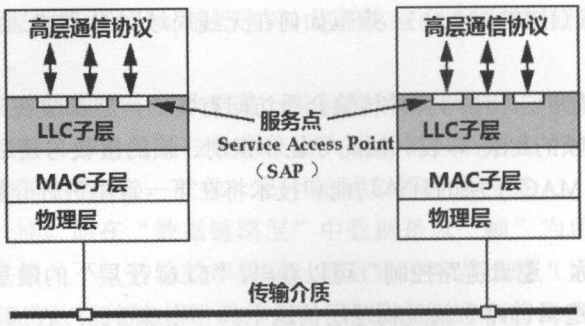


图 6-5　LLC 子层提供的服务点

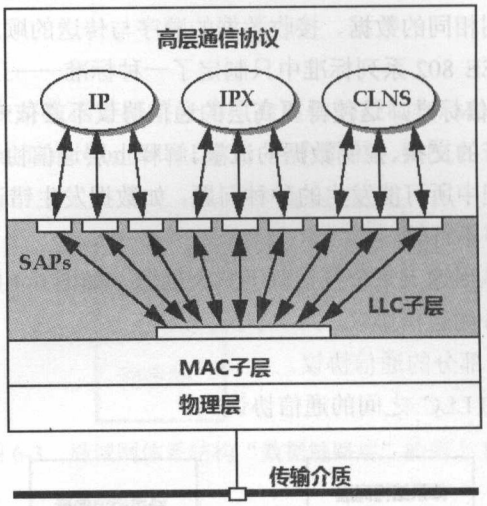


图 6-6　一种网络层通信协议可以同时使用多个服务点的示意图

为了能够辨认出 LLC 子层上传送的数据是从哪里来，要到哪里去，在 LLC 子层上传输的每个 LLC 协议数据单元（LLC Protocol Data Unit，LPDU）上都会有“目的服务访问点”（Destination Service Access Point，DSAP）和“源服务访问点”（Source Service Access Point，SSAP）这两个地址。一对 DSAP 与 SSAP 形成通信互联。由 SSAP 发送的数据经过 LLC 子层的传送送达 DSAP，反之亦然。因此 DSAP 与 SSAP 成为独立的通信连接，彼此间所传送的数据不会与其他通信连接的数据交换。当然在传送的过程中所有通信连接的数据都必须经由唯一的 MAC 信道来传送。

在计算机网络中进行的数据传输，虽然实际上是从发送端的高层一路经过数据链路层、物理层，然后再从接收端的物理层、数据链路层一直传输到对应的高层（如图 6-7 中的实线箭头所示），而从逻辑意义上看，数据是从发送端“数据链路层”到接收端“数据链路层”间的一段段逻辑链路上进行的（如图 6-7 中的虚线箭头所示），因为在物理层中传输的比特流最终还是要转换成数据帧在数据链路层中传输。

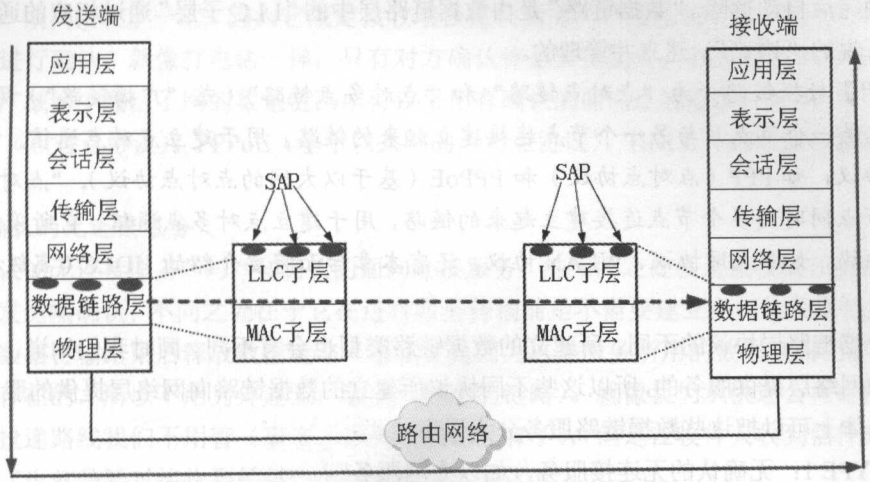


图 6-7 LLC 子层的 SAP 及数据传输原理示例

具体而言，“数据链路层”中与传输介质访问无关的问题都集中在“LLC 子层”来解决，为网络层提供服务。其主要功能包括：逻辑链路的建立和释放、提供与网络层的接口（也就是前面说到的 SAP）、数据传输差错控制、给数据帧加上传输序列号等。本章所介绍的内容主要是针对 LLC 子层（也是在没有划分 MAC 子层的网络体系结构中“数据链路层”的完全功能），MAC 子层将在下一章中专门介绍。

6.2 数据链路层主要功能及实现原理

“数据链路层”是位于网络体系结构中“网络层”的下层，所以它的一项基本功能就是向“网络层”提供透明、可靠的数据传输服务（在计算机网络体系结构中，下一层是为相邻的上一层服务的）。“透明”是指要使在“数据链路层”上所传输的数据在内容、格式及编码上都没有限制，也就是要使一些本来用于特殊用途的控制字符（具体有哪些控制字符将在本章后面介绍）也能像正常的数据一样传输，使接收端不要误认为这些字符为控制字符；可靠的传输使数据从发送端无差错地在数据链路上传输到目的接收端。

总体而言，数据链路层（其实这里主要是针对“LLC 子层”）的主要功能就是 4 个方面：数据链路管理、封装成帧、透明传输、差错控制。

6.2.1 数据链路层服务

在本章前面说到了，在数据链路层中要形成一条更有利于数据传输的“数据链路”，而不是直接利用下面物理层中建立的“物理链路”。“物理链路”在没有人为拆除时是永久存在的，而“数据链路”一般是非永久存在的（但局域网中的数据链路是永久的），仅当有数据传输时建立并存在，

在数据传输完后自动拆除。“数据链路”是由数据链路层中的“LLC 子层”通过相应的通信规程（也就是通常所说的“协议”）建立并管理的。

【说明】数据链路分为“点对点链路”和“点对多点链路”（或“广播链路”）两种。“点对点链路”就是一个节点只与另一个节点连接建立起来的链路，用于建立点对点通信。它所采用的是点对点协议，如 PPP（点对点协议）和 PPPoE（基于以太网的点对点协议）。“点对多点链路”就是一个节点同时与多个节点连接建立起来的链路，用于建立点对多点通信。它所采用的通常是点对多点协议，如以太网协议、WLAN 协议，还有本章后面将要介绍的 HDLC（高级数据链路控制）协议。

根据数据链路层协议的不同，所建立的数据链路类型也会有不同。同时我们知道，数据链路层是为上面的网络层提供服务的，所以这些不同协议所建立的数据链路向网络层提供的服务类型也有所不同。总体上可以把这些数据链路服务分为以下 4 类：

- TYPE 1：无确认的无连接服务，如以太网服务。
- TYPE 2：有确认的面向连接服务，如 PPP、HDLC 等服务。
- TYPE 3：有确认的无连接服务，如令牌网服务。
- TYPE 4：可同时提供 TYPE 1～3 类服务。

TYPE 1 和 TYPE 3 称为“无连接服务”（Connetionless Service），TYPE 2 称为“面向连接服务”（Connection-oriented Servce）。

【说明】其实不仅本章中所讲的数据链路层协议有“面向连接”和“无连接”两种服务类型，在“网络层”和“传输层”协议中也有这两种类型，如网络层中的 X.25 协议是面向连接的，而 IP 协议是无连接的；传输层中的 TCP 协议是面向连接的，而 UDP 协议是无连接的。这些将在第 10 章中进行介绍。

1. 有确认的面向连接服务

“有确认的面向连接服务”里面包括两层含义：一是在提供服务时必须先建立好双方的通信连接；二是在提供服务时必须要求对方确认后才进行。这种服务类型存在 3 个阶段：数据链路建立、数据传输、数据链路释放。举个现实中的例子，就像我们打电话，我们打电话给某个人时，首先就是要拿起电话拨号（相当于建立连接的过程），然后对方拿起电话，问一下看是不是打错了（这就是一个“确认”过程）；确认不是打错的电话后，即双方开始通话，这就相当于在数据链路中进行数据传输的过程；通话完毕，双方挂掉电话，相当于链路释放的过程。

从以上这个打电话的例子可以看出，数据链路层中“有确认的面向连接服务”是独占链路的，只有在当前数据传输完成，释放了链路后，其他用户才可能与同一个接收端进行数据传输。就像你打电话给你的朋友时，其他人再给你朋友打电话时听到的是忙音，只有等你结束了与你朋友的通话后，其他人才可以打得通你这个朋友的电话。

同样，从以上分析可以得出，“有确认的面向连接服务”非常可靠，一则是因为有专门的通信链路，在一路通信使用某条链路时，其他通信不能同时使用这条链路；再则是这种服务类型不会向错误的接收端进行数据传输，也可确认接收端正确地接收了发送来的数据，而且是按数据帧发送顺

序接收，每一帧只接收了一次，因为它规定接收端在接收到每一个数据帧（每个帧都有编号）后必须对发送端进行确认，就像打电话一样，只有对方确认你是要找他的，他才可能接听你的电话。

大多数广域网中通信子网的数据链路层协议采用有确认的面向连接服务，如 SLIP（串行线路协议）、PPP（点对点协议）、PPPoE（基于以太网的点对点协议）、HDLC（高级数据链路控制）协议等。

2. 有确认的无连接服务

“有确认的无连接服务”与“有确认的面向连接服务”的相同之处就是接收端在接收到每一个数据帧时向发送端确认；不同之处在于它在进行数据传输前是不需要建立专门的数据链路的，自然也不需要在数据传输结束后释放数据链路（事实上是因为这类服务所用的数据链路已经建立起来，而且是永久存在的，所以不用另外建立，如局域网中的链路）。就像我们从快递公司寄快递信件一样，信件的投递路线我们不用管（事实上投递路线已经有了），但是在收件人收到信件时必须要求收件人签收（也就是要对接收到的每一个数据帧进行确认）。

“有确认的无连接服务”虽然不用建立专门的连接，但仍可以保证数据的可靠传输，因为它有“确认”功能，如令牌环网和令牌总线网中的数据传输就是采用这种服务类型的，接收端在接收到一个数据帧时会发送确认信息给发送端。这类服务的另外一个主要用途就是用于一些不可靠信道中的数据传输，如各种无线通信系统。

3. 无确认的无连接服务

“无确认的无连接服务”与前面的“有确认的无连接服务”的相同之处就在于它们都是不需要在进行数据传输前先建立专门的数据链路，也就是无须先在通信双方建立通信连接；不同之处就是在进行数据传输时不要求接收端对所接收到的每一个数据帧进行确认。就像我们从邮局寄平信一样，信件投递路线我们不用管（事实上投递路线也已经有了），而且当信件到达收信人时也不用收件人签名确认。

这种服务类型看似不可靠，但它是建立在可靠的通信线路基础之上的，所以数据传输仍然是非常可靠的。如我们常用的以太网中所使用的各种以太网协议就是采用这种服务的，因为以太网中的数据链路性能非常好，数据可靠传输有保障。在以太网中的数据链路始终是存在的，不用另外建立，在以太网中进行数据传输时接收端也不用对接收到的每一帧进行确认。

6.2.2 LLC 子层服务原语

LLC 子层提供给网络层的服务是利用以下 4 种基础服务原语：

- 请求（Request）：用户实体要求服务做某项工作。
- 指示（Indication）：用户实体被告知某事件发生。
- 响应（Response）：用户实体表示对某事件的响应。
- 确认（Confirm）：用户实体收到关于它的请求的答复。

如图 6-8 所示，网络层软件经由“请求”服务原语来提出服务的要求，经过网络的传送之后到达对方。接收者的 LLC 子层则以“指示”服务原语来指示其上面的网络层软件。网络层软件获取

指示服务原语后经“响应”服务原语进行应答，经由网络传送到发送端的 LLC 子层，再以“确认”服务原语来回答发送端网络软件所提出的请求（如请求成功或失败等）。

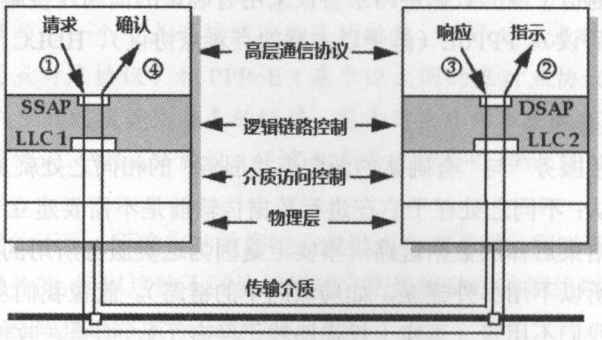


图 6-8　LLC 子层服务原语应用流程

以上只是基本的 LLC 子层服务原语应用流程，但对于不同类型的 LLC 服务，所能提供的服务原语是不完全一样的。

1. TYPE 1 服务的服务原语

在 TYPE 1 的服务中不需要建立专门的连接，而且不提供可靠的传输服务，因此没有“响应”和“确认”服务原语，其提供的服务只有两个服务原语：

- DL-UNITDATA.request（数据传输请求）
- DL-UNITDATA.indication（数据接收指示）

2. TYPE 2 服务的服务原语

在 TYPE 2 的服务中则根据所要实现的不同功能将所提供的服务原语分为以下 5 类：

（1）通信连接建立（Connection Establishment）过程中的服务原语。

- DL_CONNECT.request（连接请求）
- DL_CONNECT.indication（连接指示）
- DL_CONNECT.response（连接响应）
- DL_CONNECT.confirm（连接确认）

（2）数据传输（Data Transfer）过程中的服务原语。

- DL_DATA.request（数据传输请求）
- DL_DATA.indication（数据接收指示）

如图 6-9 所示为建立通信连接的过程范例（对应图中的序号）。

①发送端网络层软件进程利用 DL_CONNECT.request 服务原语向其下面的 LLC 子层发送连接请求，表示其欲建立一个通信连接。

②此连接请求经过网络传送到接收端的 LLC 子层，根据报文中的 DSAP，LLC 子层利用 DL_CONNECT.indication 服务原语来通知其上面对应的网络层软件进程，表示有人欲与其建立通信连接。

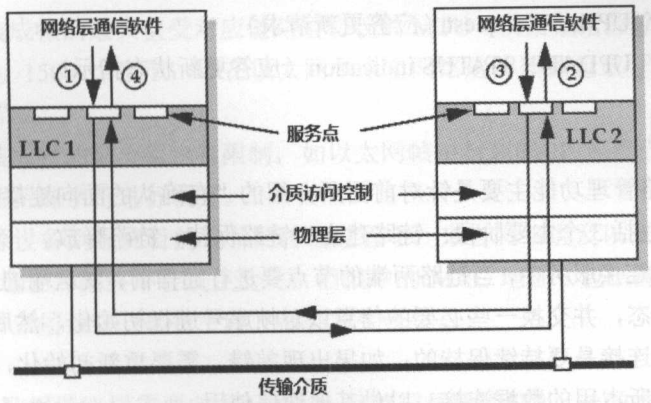


图 6-9 TYPE 2 类服务的通信连接建立流程示例

③接收端的网络层软件进程同意此请求后，再以 DL_CONNECT.response 服务原语向发送端进行响应。

④接收端的响应原语经过网络到达发送端 LLC 子层后，根据响应报文中的 DSAP 以 DL_CONNECT.confirm 服务原语向其上面对应的网络层软件进程进行确认（可能成功，也可能失败）。

（3）通信连接终止（Connection Termination）过程中的服务原语。

- DL_DISCONNECT.request（连接终止请求）
- DL_DISCONNECT.indication（连接终止指示）

（4）通信连接重置（Connection Resetting）过程中的服务原语。

- DL_RESET.request（连接重置请求）
- DL_RESET.indication（连接重置指示）
- DL_RESET.response（连接重置响应）
- DL_RESET.confirm（连接重置确认）

（5）流量控制（Flow Control）过程中的服务原语。

- DL_CONNECTION_FLOWCONTROL.request（连接流量控制请求）
- DL_CONNECTION_FLOWCONTROL.indication（连接流量控制指示）

3. TYPE 3 服务的服务原语

TYPE 3 服务提供的是不建立通信连接（无连接）但有回复确认的传输服务，其提供的服务原语有以下 8 个：

- DL_DATA_ACK.request（数据确认请求）
- DL_DATA_ACK.indication（数据确认指示）
- DL_DATA_ACK_STATUS.indication（数据确认状态指示）
- DL_REPLY.request（应答请求）
- DL_REPLY.indication（应答指示）
- DL_REPLY_STATUS.indication（应答状态指示）

- DL_REPLY_UPDATE.request（应答更新请求）
- DL_REPLY_UPDATE_STATUS.indication（应答更新状态指示）

6.2.3 数据链路管理

LLC 子层的链路管理功能主要是针对前面所介绍的“有确认的面向连接服务”类型（主要应用于广域网中）。它包括 3 个主要阶段：链路建立、链路保持、链路释放。

在有确认的面向连接服务中，当链路两端的节点要进行通信前，发送端的数据链路层必须先确认对方已处于就绪状态，并交换一些必要的信息以对帧序号进行初始化，然后双方才能建立连接；在传输过程中各数据连接是要持续保持的；如果出现差错，需要重新初始化，重新自动建立连接。传输完毕后则要释放所占用的数据连接，以供其他通信使用。

数据连路层的这种链路连接的建立、维持和释放过程就是数据链路层的链路管理功能。在多个站点共享同一物理信道的情况下（例如在局域网中），如何在要求通信的站点间分配和管理信道也属于数据链路层管理的范畴。

6.2.4 数据帧封装与解封装

在发送端，数据链路层是接收来自网络层的数据，而在接收端，数据链路要向网络层上传数据。这里面就涉及到数据链路层的两个相反作用的功能，即“封封装”（在发送端）和“帧解封装”（在接收端）。下面分别予以介绍。

1. 帧封装

数据链路层的“帧封装”功能是指将来自网络层的数据包加装对应数据链路层协议头部和尾部封装成数据帧的过程。在每个帧的前部加上一个“帧头部”，在帧的结尾处加上一个“帧尾部”，把网络层的数据包作为帧的数据部分就构成了一个完整帧。帧头是数据链路层协议的头部，帧尾是帧校验序列（FCS）字段。帧头和帧尾就是作为帧的起始和结束标志，也就是帧边界，如图 6-10 所示。

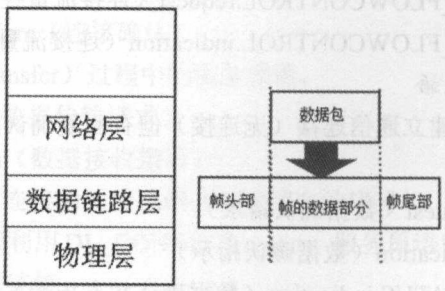


图 6-10 帧封装示意图

在帧封装过程中，来自网络层的所有数据都被当作成帧中的“数据”字段部分，但需要注意以下几个方面：

- 数据包封装成帧的最大值受对应链路层协议的 MTU（仅指数据部分大小）限制，如以太网中最大为 1500 个字节，即在以太网中来自网络层的数据包或分组大小最大不能超过 1500 个字节。
- 数据包封装成帧的最小值也有限制，如以太网帧中封装的 IP 包最小值为 46 个字节，否则要用一些特殊字符进行填充。
- 数据包从源设备到最终目的设备的传输过程中，来自网络层及以上各层的数据（包括源和目的 IP 地址）始终不会发生变化，但帧中的 MAC 地址（包括源和目的 MAC 地址）每经过一个三层设备（即每一跳）都会发生变化。

2. 帧解封装

“帧解封装”是数据链路层需要向上一层——网络层传输数据时（该数据一定是来自发送端的网络层或者更高层），需要把原来数据帧中的“帧头部”和“帧尾部”去掉（仅保留帧中的“数据”字段部分），还原前面在发送端从网络层发送时对应的数据包或数据分组格式再向上层传输。基本流程如图 6-11 所示。

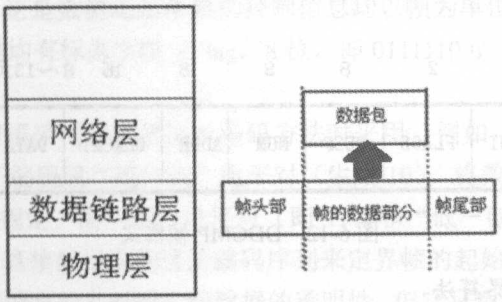


图 6-11 帧解封装示意图

6.2.5 帧组装与帧同步

在发送端从数据链路层发送的帧需要经过物理层转换成一个个比特进行传输，但到了接收端它又需要向上面的数据链路层传输，这时就涉及到数据链路层的另两个功能：帧组装和帧同步。

1. 帧组装

在发送端“数据链路层”中的帧到达“物理层”后就会以“比特位”为单位进行传输，而不是以“帧”为单位进行传输，尽管在并行传输方式中可以一次传输一个或多个字节，但每条线路中的传输单位还是“比特位”。到了接收端后，物理层收到的比特流需要继续向它的上一层——数据链路层传输，此时需要把这些比特流组装成与发送端对应数据一样的帧格式，这就是数据链路层的“帧组装”功能。

在“帧组装”的过程中涉及到“帧同步”功能，即在接收端如何区分来自发送端的比特流属于原来在发送端中的哪个帧。当然，这需要在发送端进行数据帧发送时就确定好，然后在接收端依据

这些帧同步方法进行帧组装。常见的帧同步方法有：字节计数法、字符填充的首尾定界符法、比特填充的首尾定界符法、违法编码法。

2. 字节计数法

这是一种以一个特殊字符代表一个帧的起始，并以一个专门的字段来标识当前帧中“数据”字段部分字节数的帧同步方法。接收端可以通过对所接收的比特流中的该特殊字符的识别区分出每个帧的起始，并从专门字段中获知每个帧后面跟随的“数据”字段的字节数，从而可以确定出每个帧的结束位置。

这种面向字节计数的同步规程的典型实例是DEC公司的DDCMP(Digital Data Communications Message Protocol，数字数据通信报协议)。在DDCMP协议帧格式（如图6-12所示）中，SOH字段是一个帧的帧头开始部分，有其固定的值（十进制值为129，八进制值为201，十六进制值为81），就相当于一个帧开始的特殊字符；在NUM字段中为每个数据帧分配一个编号，从“1”开始，并以“1”为增量进行递增，最大值为256（也就是模为256），以确保在从站点中的正确消息序列，同时在COUNT字段中指出本数据帧中DATA（数据）字段的大小，这些都是用来进行帧同步的。

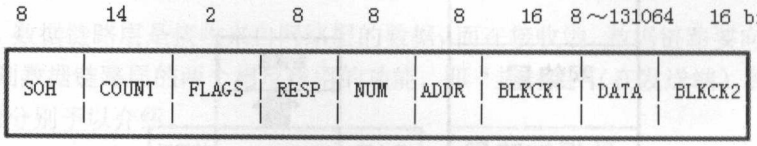


图6-12　DDCMP帧格式

3. 字符填充的首尾定界符法

该同步方法是用一些特定的控制字符来定界一个帧的起始与结束，如IBM的BSC协议在每个数据块的头部用一个或多个同步字符SYN来标记数据块的开始，尾部用字符ETX来标记数据块的结束。

如图6-13所示是要传输一个“ADFGJ”字符串，在帧的头部加上了两个SYN控制字符，用于标识该帧的开始；在结束位置加上了ETX控制字符，用于标识该帧的结束。

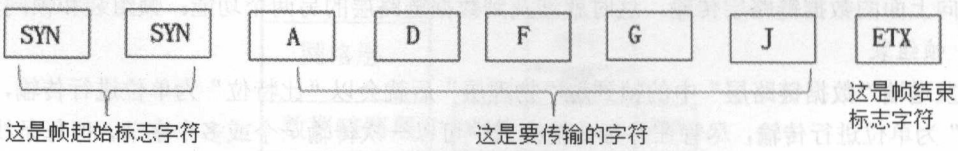


图6-13　字符填充的首尾定界符帧同步方法应用示例

另外，为了不使数据信息中与以上特定字符相同的字符被误判为帧的首尾定界符，可以在这种数据帧的帧头填充一个转义控制字符（DLE，Data Link Escape – Start of Text），这就属于“透明传输”原理，将在本章后面介绍具体的数据链路层协议时介绍。

4. 比特填充的首尾定界符法

该帧同步方法是通过在帧头和帧尾各插入一个特定的比特串（如 01111110）来标志一个数据帧的起始与结束，这个帧头、帧尾特定比特串称为“帧标志符”。如传输的比特流为 1001110101，组装成帧后就是 01111110100111010101111110。

为了“透明传输”，也就是为了避免在信息位中出现与帧起始和结束标志符相似的比特串时被误判为帧的首、尾标志，采用了比特填充的方法。比如上面采用的特定模式为 01111110，则对信息位中的任何连续出现的 5 个“1”（因为帧标志符中是有 5 个连续的“1”），发送端自动在其后插入一个“0”。如要传输的数据帧为 0110111111011111001（因为其中有 5 个连续的“1”，很可能被误认为是帧首、尾标志），采用比特填充后，实际传送的是 01111110011011111*0*10111110*0*01011111110（注意，前、后面两个 01111110 是帧首、尾标志符），在有 5 个连续“1”的比特位后面加插入一个“0”（斜体“*0*”）。而接收方在收到上述最终数据后进行发送端的逆操作，首先去掉两端的特定模式字符，然后在每收到连续 5 个“1”的比特位后自动删去其后所跟的“0”，以此恢复原始信息。

比特填充帧同步方式很容易由硬件来实现，性能优于字符填充方式。所有面向比特的同步控制协议采用统一的帧格式，不论是数据还是单独的控制信息均以帧为单位传送，其典型代表是 ISO 的 HDLC 协议，在它的首尾均有标志字段（Flag，8 位，即 0111110），具体将在本章后面介绍。

5. 违法编码法

该帧同步方法是在物理层采用特定的比特编码方法时采用。例如，曼彻斯特编码方法（参见 4.6.4 节），将数据比特“1”编码成“高－低”电平对（生成 10），将数据比特“0”编码成“低－高”电平对（生成 01）。并规定“高－高”电平对（即 11）和“低－低”电平对（即 00）在数据比特中是违法的，因此可以直接借用这些违法编码序列来定界帧的起始与终止。

违法编码法不需要任何填充技术便能实现数据的透明性，但它只适用于采用冗余编码（也就是原来一位信号码经过编码后变成了多位）的特殊编码环境。

6.2.6 差错控制

【说明】因“数据链路层”中的差错控制功能涉及到比较多且比较复杂的技术，所以在此仅进行综合介绍，具体的差错控制技术将在本章后面介绍。

在上节介绍的“成帧”功能解决了帧同步问题，也就是接收端可以区分每个数据帧的起始和结束了，但是还没有解决数据正确传输的两方面问题：一是如果有帧出现了错误怎么办；二是如果有帧丢失了怎么办。这都是“数据链路层”确保向网络层提供可靠数据传输服务要解决的问题，也就是数据链路层的“差错控制”功能。

要实现差错控制功能，就必须具备两种能力：一是发现差错的能力，二是纠正错误的能力。就像我们要发现一个“问题”小孩一样，你首先要知道他的“问题”在哪里，也就是存在问题的根源在哪里，然后才能采取适当的方法纠正这个问题小孩身上的问题。

1. 差错检测

在数据链路层检测数据传输错误的方法一般是通过对差错编码进行校验来实现，常见的校验方

法有奇偶校验码（PCC）、循环冗余校验（CRC）两种。它们都称之为“检错码”（error-detecting code）。

“奇偶校验法”（Parity Check Code，PCC）是一种校验代码传输正确性的方法，是根据被传输的一组二进制代码的数位中“1”的个数是奇数或偶数来进行校验。采用“1”的奇数个校验的方法称为“奇校验”，反之称为“偶校验”，但采用何种校验是事先确定好的。

具体做法是在传输的二进制代码最后专门设置一个奇偶校验位，用它使这组代码中“1”的个数为奇数或偶数（具体是偶数还是奇数要视所采用的是偶校验还是奇校验），然后再在接收端进行校验，看里面的“1”的个数是否仍与原来一样的奇数或偶数来确定数据传输的正确性。

循环冗余校验（Cyclic Redundancy Check，CRC）是一种根据传输或保存的数据而产生固定位数校验码的一种散列函数，主要用来检测或校验数据传输或者保存后可能出现的错误。生成的数字在传输或者储存之前计算出来并且附加到数据后面，然后接收端进行检验确定数据是否发生变化。

2. 差错纠正

在差错纠正方面，数据链路层针对不同的传输类型采取了不同的纠错方法。对于面向字符的异步传输（如键盘与主机的通信、ATM 传输协议等）一般是采用“反馈检测”的方法来进行纠错。就是在接收端收完一帧数据后，向发送端发送回所接收到的完整数据帧，由发送端通过与原始发送的帧进行比较来判断接收端是否正确接收了对应帧。如果判断是出了错误，则发送端向接收端发送一个 DEL 字符及相应的帧信息，提示接收端删除对应的帧，然后重发该帧；否则表示接收端已正确接收了对应的帧，不重发对应的帧。但对于在传输过程中完全丢失的数据就不能采用这种纠错方法了，因为接收端根本没有收到这帧数据，所以也就不会向发送端发回反馈信息，发送端自然就不能确认接收端是否正确接收了对应的帧。

为了解决这个问题，通常在数据发送时引入计时器（Timer）来限定接收端发回反馈信息的时间间隔。当发送端发送一帧的同时也启动计时器，若在限定时间间隔内没有收到接收端的反馈信息，即计时器超时（Timeout），则可认为传的对应帧已出错或丢失，继而发送端知道要重新发送对应的数据帧。同时，为了避免同一帧数据可能被多次重复发送，引发接收端多次收到同一帧并将其递交给网络层的危险，采用对发送的帧进行编号的方法，即同一个帧的编号是一样的，从而使接收端能从帧编号来区分是新发送来的帧还是已经接收但又重新发送来的帧，以此来确定要不要将重新接收到的帧递交给网络层。

【注意】并不是每种帧都有“帧编号”字段，主要是在广域网中的 SDLC（Synchronous Data Link Control，同步数据链路控制）和 HDLC（High Level Data Link Control，高级数据链路控制）协议帧中，在以太网和 PPP 协议帧中是没有这个字段的，也就不能采用这种差错控制方法，但因为这两种网络传输的可靠性较高，所以并不影响数据传输的可靠性。这两种协议将在本章后面介绍。

由于在“反馈检测法”中一帧数据会在信道中至少往返传输两次，传输效率低，所以一般不会采用这种差错控制方法，而是采用一种称为“自动重发请求”（ARQ）的方法。实现原理就是先让发送端将要发送的数据帧附加一定的冗余检错码（如前面介绍的 PCC 码、CRC 码）一起发送，接收端则根据检错码对数据帧进行错误检测，若发现错误，就返回请求重发的响应（不用返回全部的帧），发送端收到请求重发的响应后便重新传送该数据帧。

另外，还有一些编码本身具有自动纠正错误的能力，称为“纠错码”（Error-Correcting Code），在数据链路层中常用的如海明码（Hamming Code）。

以上差错控制方法都将在本章后面具体介绍。

6.2.7 流量控制

“流量控制”包括两方面的含义。一是发送端的数据发送速度与接收端的数据接收速度要匹配，否则接收端来不及接收就会造成数据在传输过程中的丢失。这个很好理解，比如我们几个人站在一排传递货物时，如果中间有个人速度比较慢，而前面的人不断传来货物，肯定就会把来不及接收的货物放在地上（不进入正常的传递之中）。二是发送端的数据发送速度要与线路上的承载速率（与线路信道带宽有关）相匹配，否则也会造成数据在传输过程中的丢失。这个也很好理解，就像一条小河，上游来的水量很大，超过了小河所能承载的能力，这时上游来的水肯定就不会有原来那么大的流速（形成速率瓶颈），甚至可能会漫过小河河堤而流到外面去了。

【注意】“流量控制”并不是数据链路层所特有的功能，许多高层协议中也提供流量控制功能，只不过流量控制的对象不同而已。比如，对于数据链路层来说控制的是相邻两节点之间数据链路上的流量，使链路两端的传输速率尽可能匹配；对于网络层来说是控制不同网络间的流量，尽量保持匹配；而对于“传输层”来说控制的则是从源到目的端之间的流量，使发送端和最终的接收端之间的传输速率尽可能匹配。

在数据通信中，由于通信双方各自使用的设备工作速率和缓冲存储空间的差异，可能出现发送端发送能力大于接收端接收能力的现象，如果此时不对发送端的发送速率作适当的限制，在接收端前面来不及接收的帧将被后面不断发送来的帧所“淹没”，从而造成帧的丢失。由此可见，数据链路层上的“流量控制”功能实际上是对发送端数据传输速率的控制，使其数据发送速率不超过链路对端接收端口所能承受的数据接收能力。考虑到在接收端还需要对来自物理层的比特流进行一系列的处理，如帧封装、向发送端发送返回确认帧等，所以通常是要使发送端的发送速率略小于接收端的数据处理能力。

在数据链路层中进行流量控制主要有两种方案：一是基于反馈的流量控制方案，二是基于速率的流量控制方案。

1. 基于反馈的流量控制方案

“基于反馈的流量控制方案”就是接收端在接收到一个数据帧后要向发送端发送一个确认帧，表示发送端可以继续向它发送数据了。这种方案也称“停－等”方案（也就是6.3.4节中介绍的“空闲重发请求”方案），就是发送端在发送一帧数据后必须等待接收端返回确认响应消息，然后才能发送下一数据帧。接收端检查帧的校验序列（FCS），无错则发送确认帧，否则不发送返回消息，表示该帧已出错，要求重发。

其实这也就是利用上节介绍的差错控制方案。但这里存在一个问题，就是在数据帧或确认帧丢失时双方会无休止等待。解决这一问题的方法是发送后使用定时器，在发送端发送一帧数据时会启动这个定时器，要求接收端必须在这个时间内返回确认帧，否则就认为这个数据帧已丢失，重发该

数据帧。当然，与上节介绍的差错控制方案时提到的一样，这也可能造成接收端接收到多个一样的数据帧，解决这一问题的方法就是给每个帧加上一个编号，这样编号一样的帧接收端只需接收其中一个即可，其他的丢弃。

由此可以看出，“停－等”方案其实就是直接利用了上节介绍的纠错方案，这里提到的几种措施可以说是一环扣一环，最终可以既确保数据的可靠传输（也就是“差错控制”功能），又有效地控制了双方通信的流量（也就是“流量控制”功能）。

2. 基于速率的流量控制方案

“基于速率的流量控制方案”是基于窗口滑动机制的速率控制方案，它规定发送端一次可以发送多少个数据帧，限制了发送端的数据传输速率，而无需接收端发回确认帧。这种机制在本章后面也有专门介绍。其实它与第 10 章将要介绍的传输层流量控制方案是极其类似的，不同的只是数据链路层的流量控制是针对链路两端的点对点控制，而传输层则是直接针对通信双方的端到端系统。

6.3 差错控制方案

在本章前面已经说到，“数据链路层”的差错控制功能分为“差错检测”功能和“差错纠正”功能。在差错检测方面通常是在数据帧中加上一些具有检错功能的冗余代码（称为“检错码”（Error-Detecting Code）），然后根据这些冗余代码所表达的信息以及接收端对数据帧的奇偶重新计算结果就可以发现数据帧在传输过程中是否出现差错。本章前面说到的 PCC（奇偶检验码）、CRC（循环冗余校验）都属于“检错码”类型。但它们均不具有自动纠错功能，不能确定是哪一个或哪一些位是错误的，也不能纠正这些差错，通常也是与纠错方案中的“自动请求重发法”（ARQ 法）结合起来进行差错控制的。

在差错纠错方面主要有“反馈检测法”和“自动重发请求法”两种（“自动重发请求法”又分为几种）。也有一些具有自动纠错的编码，称为“纠错码”（Error-Correcting Code），又称为“前向纠错码”（Forward Error Correction），如海明码（Hamming Code）。这些方案都将在下面各小节中具体介绍。

6.3.1 奇偶校验码检错方案

“奇偶校验码”（PCC）是“奇校验码”和“偶校验码”的统称，是一种有效地检测单个错误的检错方法。它的基本校验思想是在原信息代码的最后添加一位用于奇校验或偶校验的代码（可为 0 或 1），这样最终的帧代码是由 n-1 位信元码和 1 位校验码组成的，可以表示成为(n,n-1)。

加上校验码的最终目的就是要让传输的帧中“1”的个数固定为奇数（采用奇校验时）或偶数（采用偶校验时），然后通过接收端对接收到的帧中“1”个数的实际计算结果与所选定的校验方式进行比较，就可以得出对应帧数据在传输过程中是否出错了。如果是“奇校验码”，在附加上一个校验码以后，码长为 n 的码中“1”的个数为奇数；如果是“偶校验码”，在附加上一个校验码以后，

码长为n的码中“1”的个数为偶数（0个“1”也看成是偶数个“1”）。“奇偶校验”方法可以通过电路来实现，也可以通过软件来实现。

下面打个很简单的比喻来说明PCC的校验原理。现假设一学校要组织各班学生参加一个长跑运动，同时有几名外地学生想参加长跑运动，需要在每班插入一个外地学生，但插入的学生性别不能随意，要求插入外地学生后各班学生的女生数必须为偶数（相当于采用偶校验方式，当然也可以要求必须为奇数，则此时为奇校验方式），这样就可确定每班插入的这位外地学生的性别了。如果长跑后各班清点人数时发现有些班的女生数量不是原来规定的偶数（或奇数），则证明这些班的学生在长跑过程中走乱了（相当于数据在传输途中出现了差错）或者有学生掉队了（相当于数据在传输途中丢失了）。这就是“奇偶校验”原理。

我们知道ASCII字符是8位编码，其中低7位是信息代码，最高一位是奇偶校验位。如现在传输的是ASCII字符，如果采用奇校验，则每个ASCII字符代码中“1”的个数均必须为奇数个，如果发现某个字符中“1”的个数是偶数，则这个ASCII字符在传输过程中肯定出错了；同理，如果采用的是偶校验，则每个ASCII字符代码中“1”的个数均必须为偶数个，如果发现某个字符中“1”的个数是奇数，则这个ASCII字符在传输过程中也肯定出错了。其他采用奇偶校验方式的数据的校验原理也是一样的。

假设现在要传输一个ASCII字符，它的低7位代码为1011010，如果现在要采用奇校验方法，则该字符的校验码为“1”，整个ASCII字符代码就是**1** 1011010，因为该字符中低7位信息代码中的“1”的个数是偶数个（4个），必须再加一个“1”才能为奇数；同理，如果要采用偶校验方法，则该字符的校验码为“0”，整个ASCII字符代码就是**0** 1011010，因为该字符中低7位信息代码中的“1”的个数已经是偶数个（4个），所以最高位不能再是“1”，只能是“0”。

在这里要注意，无论采用的是奇校验方法还是偶校验方法，每一信息的校验位是“0”还是“1”都是不固定的，要视具体信息前n-1位中“1”的个数而定。如果采用的是“奇校验”，则要求所传输的信息中n位中“1”的总个数必须为奇数；如果前面n-1位中“1”的个数已经是奇数，则第n位的校验码位一定是“0”，只有这样才能确保整个n位信息中“1”的个数仍为奇数；如果前面n-1位中“1”的个数是偶数，则第n位的校验码位一定是“1”，只有这样才能确保整个n位信息中“1”的个数为奇数。

如果采用的是“偶校验”，则要求所传输的信息中n位中“1”的总个数必须为偶数（0个也看成是偶数个）；如果前面n-1位中“1”的个数是奇数，则第n位的校验码位一定是“1”，只有这样才能确保整个n位信息中“1”的个数为偶数；如果前面n-1位中“1”的个数是偶数，则第n位的校验码位一定是“0”，只有这样才能确保整个n位信息中“1”的个数仍为奇数。

另一个要注意的是，奇偶校验方法只可用来检查单个码元错误，检错能力较差，所以一般只用于本身误码率较低的环境，如以太局域网中，还用于磁盘中的数据存储。如现在采用的是奇校验方法，传输的整个8位ASCII字符代码为**1** 0100100（最高位为校验位），如果里面低7位信息代码有一位出现了差错，由原来的“0”变为“1”或者由原来的“1”变为“0”，都会改变里面“1”的个数，最终导致与对应的奇校验方法不一致，在这种情况下PCC能判断这个字符传输出错。但是

如果里面前 7 位信息代码同时有两位或多位出现了差错，最终的结果可能会使字符的整个 8 位代码中“1”的个数奇偶性不变，而不能判断这个字符传输出错。如以上字符代码变为 1 0001100，结果“1”的个数仍为奇数个（3 个），但实际上该数据已不是原来的数据了。这就是为什么奇偶校验方法只能检测出单个码元错误的原因。

作为课后练习，大家自己再分析一下，如果所传输的二进制序列是 11101100101，现要采用奇校验，则校验位的值是什么？采用偶校验呢？

6.3.2 循环冗余校验检错方案

上节介绍的奇偶校验码（PCC）只能校验一位错误，本节所要介绍的循环冗余校验码（CRC）的检错能力更强，可以检出多位错误。在以太网帧后面的 FCS（帧校验序列）通常就是采用 CRC 的校验码。

1. CRC 校验原理

CRC 校验原理看起来比较复杂，好难懂，因为大多数书上基本上是以二进制的多项式形式（就像我们在第 1 章中介绍二进制数转换为十进制数那样的展开式）来说明的。其实是很简单的问题，其根本思想如下：

（1）在发送端在发送的帧后面附加一个数（冗余码，但是要注意，这里的数也是二进制序列的，不是一位二进制码），生成一个新帧发送给接收端。

（2）接收端再把接收到的新帧除以（同样采用“模 2 除法”）这个冗余码。如果有余数，则数据帧在传输中出现了差错。

当然，以上这个附加的数不是随意的，它要使所生成的新帧能与发送端和接收端共同选定的某个特定数整除（注意，这里不是直接采用二进制除法，而是采用“模 2 除法”）。到达接收端后，再把接收到的新帧除以（同样采用“模 2 除法”）这个选定的除数。因为在发送端发送数据帧之前就已通过附加一个数做了“去余”处理（也就是已经能整除了），所以结果应该是没有余数的。如果有余数，则表明该帧在传输过程中出现了差错。

【说明】“模 2 除法”与“算术除法”类似，但它既不向高位借位，也不比较除数和被除数的相同位数值的大小，只要以相同位数进行相除即可。

模 2 除的过程要采用“模 2 减法”运算，法则为：1-1=0，0-1=1，1-0=1，0-0=0，也无进位，无借位，相当于二进制中的逻辑异或运算。也就是比较后，两者对应位相同则结果为“0”，不同则结果为“1”。如 100101 除以 1110，结果得到商为 11，余数为 1，如图 6-14 所示。

2. CRC 校验码计算步骤

具体来说，CRC 校验码的计算步骤如下：

（1）选择（可以随机选择，也可按标准选择，具体在后面介绍）一个用于在接收端进行校验时对接收的帧进行除法运算的除数（是二进制比特串，通常是以多项式表示，所以 CRC 又称多项式编码方法，这个多项式也称为“生成多项式”）。

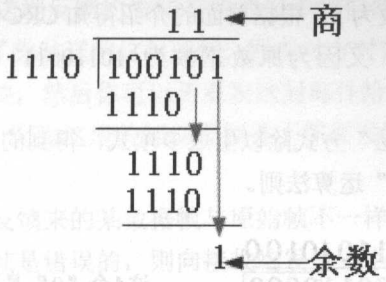


图 6-14 “模 2 除法”示例

（2）看所选定的除数二进制位数（假设为 k 位），然后在要发送的数据帧（假设为 m 位）后面加上 k-1 位“0”。

（3）以这个加了 k-1 个“0“的新帧（一共是 m+k-1 位）以“模 2 除法”方式除以上面这个除数，得到的余数（也是二进制的比特串）就是该帧的 CRC 校验码，也称为 FCS（帧校验序列）。但要注意的是，余数的位数一定要比除数的位数只能少一位，哪怕前面位是 0，甚至是全为 0（正好整除时）也都不能省略。

（4）把这个校验码附加在原数据帧（就是 m 位的帧，注意不是在后面形成的 m+k-1 位的帧）后面，构建一个新帧发送到接收端，在接收端再把这个新帧以“模 2 除法”方式除以前面选择的除数，如果没有余数，则表明该帧在传输过程中没有出错，否则出现了差错。

通过以上介绍的这 4 步，大家一定可以理解 CRC 校验的原理，并且不再认为很复杂了。就两个关键点：一是要预先确定一个发送端和接收端都用来作为除数的二进制比特串（或多项式）；二是把原始帧与上面选定的多项式进行模 2 除法运算，计算出 CRC 校验码。

多项式可以随机选择，也可按国际上通行的标准选择，但最高位和最低位必须均为“1”，而且两端必须统一。如在 IBM 的 SDLC（同步数据链路控制）规程中使用的 CRC-16（也就是这个除数一共是 17 位）生成多项式 $g(x)= x^{16} + x^{15} + x^2 + 1$（对应二进制比特串为 11000000000000101）；而在 ISO HDLC（高级数据链路控制）规程和 ITU 的 SDLC、X.25、V.34、V.41、V.42 等中使用 CCITT-16 生成多项式 $g(x)= x^{16} + x^{15} + x^5 + 1$（对应二进制比特串为 11000000000100001）。

3. CRC 校验码的计算示例

由以上分析可知，既然除数是随机的或者按标准选定的，所以 CRC 校验的关键是如何求出余数，也就是校验码（CRC 校验码）。

下面以一个例子来具体说明整个过程。现假设选择的 CRC 生成多项式为 $g(x)= x^4 + x^3 + 1$，要求出二进制序列 10110011 的 CRC 校验码。具体的计算过程如下：

（1）把生成多项式转换成二进制数，由 $g(x)= x^4 + x^3 + 1$ 可以知道，它一共是 5 位（总位数等于最高位的幂次加 1，即 4+1=5），然后根据多项式各项的含义（多项式只列出二进制值为 1 的位，也就是这个二进制数的第 4 位、第 3 位、第 0 位的二进制均为 1，其他位均为 0）很快就可以得到它的二进制比特串为 **11001**。

（2）因为生成多项式的位数为5，根据前面的介绍得知CRC校验码的位数为4（校验码的位数比生成多项式的位数少1）。又因为原数据帧为10110011，在它后面再加4个0，得到101100110000。

（3）把这个数以“模2除法”方式除以生成多项式，得到的结果为0100，如图6-15所示。注意参考前面介绍的“模2除法”运算法则。

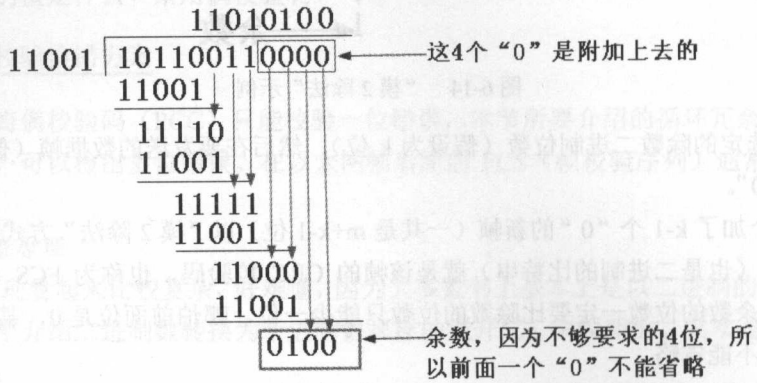


图6-15　CRC校验码计算示例

（4）把上步计算得到的CRC校验码替换帧101100110000后面的4个“0”，得到新的帧101100110100。再把这个新帧发送到接收端。接收端会把这个新帧再用上面选定的除数11001以“模2除法”方式去除，验证余数是否为0，如果为0，则证明该帧数据在传输过程中没有出现差错，否则出现了差错。

通过以上CRC校验原理的剖析和CRC校验码的计算示例的介绍，大家应该对这种看似很复杂的CRC校验原理和计算方法比较清楚了。

下面大家做一个练习，假设CRC生成多项式为$g(x)= x^5 + x^4 + x + 1$，要发送的二进制序列为100101110，求CRC校验码是多少？

6.3.3　反馈检测法

“反馈检测法”是一种最简单、最容易实现，但并不常用的差错控制方法，因为在这种差错控制方案中每个数据帧均要求至少在信道中往返传输两次，信道利用率低。

1. 反馈检测法基本原理

“反馈检测法”不需要利用前面两节介绍的PCC（奇偶校验码）和CRC（循环冗余校验码）检测错码技术，把差错检测和差错纠正的任务全部交给发送端自己。它是要求接收端在接收到每一个数据帧后均要向发送端发送一个表示是否接收了该数据帧的反馈信息，且这个反馈信息就是原来由发送端发给接收端的原始数据帧。发送端在收到接收端发送的反馈信息后，通过对比保存在缓存中原来该帧的数据来判断接收端是否正确接收了该数据帧。

这种要求接收端必须向发送端反馈的做法有些类似于我们向某人发送一个需要对方确认的重要邮件一样，如果收件人收到了你发送的这封邮件，要求对方对你进行回复确认；如果对方收到的邮件已遭破坏，也可以向你反映，然后你可以再重发这封邮件给他，直到对方收到了正确的邮件。当然这两者的机制不完全等同，因为这里的邮件反馈并不需要发回原来邮件的全部内容，只是一条表示收到了邮件的反馈信息。

如果经过比较发现接收端反馈来的某数据帧与原始帧不一样，则表明对应帧在传输过程中出现了差错，接收端接收到的对应帧是错误的，则向接收端发送一个 DEL 字符和相应的帧编号（下面将介绍到），通知接收端删除对应的帧，重发对应的数据帧，直到接收到反馈来的数据对应帧与原来发送的该帧数据完全一样；如果经比较接收端返回来的某帧与原来发送的对应帧完全一样，则表示接收端已正确接收到该数据帧，不再重发，继续发送一个数据帧。

2. 两项附加技术

从以上反馈检测原理可以发现有一个问题，那就是可能因一些原因导致一些数据帧在传输过程中完全丢失了，接收端也根本没收到这些数据帧，自然也就不会向发送端发送对应数据帧的反馈信息了。就像你要求收件人在收到你发送的邮件时给你回信，但对方根本没收到你的邮件，自然也就不会有回信了，你还总在那里等他的回信。

为了避免出现这种情况，通常引入计时器（Timer）来限定接收端发回反馈消息的时间间隔，即当发送端发送一帧数据时即自动启动计时器。如果在计时器中限定的时间间隔内仍没有收到接收端发来的反馈信息，则可认为该帧的数据传输出现了差错或丢失，就主动重新发送该帧。就像你对某人发送邮件前就告诉对方，“我现在就发送邮件给你，你必须在 2 小时内回复我是否收到了这封邮件，否则我认为你没收到这封邮件，会重新发一封给你。”

以上计时器方案虽然解决了由于数据丢失而引发的意外，但可能由于网络线路比较忙，送达的某帧数据在到达接收端时有所延迟，超出了计时器限定的时间范围，发送端仍然会重发对应的数据帧，这样就可能使接收端多次接收同一数据帧了。为了防止发生这种现象，在“数据链路层”又采取了“帧编号”（也就是“帧序列号”）方法对发送的每个数据帧进行编号（同一数据帧编号一样，无论发送多少次），从而使接收端能从该编号来区分是新发来的帧还是已经接收但又重新发来的帧，从而确定要不要将接收到的帧递交给网络层。

数据链路层的“反馈检测法”通过使用“计数器”和“帧编号”两种措施来保证每个数据帧最终都能被正确地递交给目标网络层一次。

6.3.4 空闲重发请求方案

从上节介绍的“反馈检测法”差错控制原理可以看出，虽然原理简单，容易实现，也有较高的可靠性，但每个数据帧实际上均被在信道中传输了两次，造成信道利用率降低。也正因为如此，这种差错控制方法一般用于面向字符的异步传输中，因为这种情形下，传输的数据量比较小，信道传输效率并不是主要问题。

实用的差错控制方法，要求既要传输可靠性高，又要信道利用率高。为此可使发送端将要发送

的数据帧附加一定的冗余检错码（如 PCC、CRC 等）一并发送，接收端则根据检错码对数据帧进行差错检测，若发现错误，就返回请求重发的响应，发送端收到请求重发的响应后便重新传送该数据帧。这种差错控制方法就称为“自动重发请求法”（Automatic Repeat reQuest），简称 ARQ 法。ARQ 差错控制方案仅需返回少量控制信息（接收端不必重传整个数据帧），便可有效地确认所发数据帧是否正确被接收。

ARQ 法有几种具体的实现方案，“空闲重发请求”（Idle RQ）和“连续重发请求”（Continuous RQ）是其中最基本的两种方案。本节先介绍“空闲重发请求”差错控制方案，下节再介绍“连续重发请求”差错控制方案。

“空闲重发请求”方案也称“停一等”（Stop and Wait）法，该方案规定发送端每发送一帧后就要停下来，然后等待接收端发来的确认信息（这就是“停一等”的意思），仅当发送端收到接收端发来的确认（ACK）信息后才继续发送下一数据帧，如果收到的是否认（NAK）消息，表示接收端接收的数据有错，发送端会重发该帧。另外，在计时器超时时，发送端也会重发对应的帧。如图 6-16 所示分别是正确接收数据时的数据帧发送流程和接收到有错误数据时的数据帧发送流程。

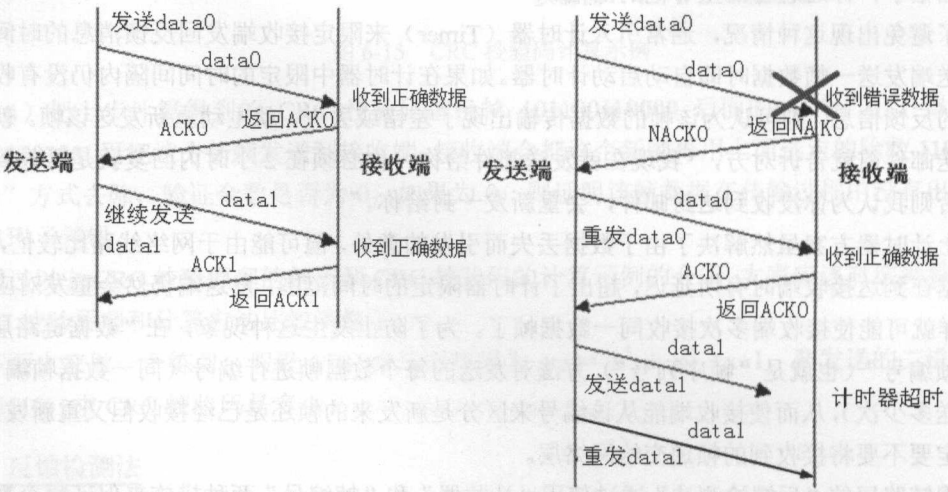


图 6-16　空闲重发请求方案原理示例

“空闲重发请求”差错控制方案的具体实现过程如下：

（1）发送端每次仅将当前数据帧作为待确认帧保留在缓冲存储器中，当发送端开始发送数据帧（如图 6-16 左、右图所示的 data0）时，随即启动计时器；

（2）当接收端收到这个数据帧时，先利用帧中附带的校验码进行校验，确认无差错后，即向发送端返回一个确认信息（如图 6-16 左图所示的 ACK0、ACK1 和右图所示的 ACK0）；当检测到该帧有错误时，向发送端返回一个否认帧（如图 6-16 右图中的 NAK0），同时丢弃该帧；

（3）如果发送端在计时器中规定的时间内收到来自接收端的确认信息，即将计时器清零，清

除缓存中的待确认帧，然后才开始下一数据帧（如图 6-16 左图所示的 data1）的发送；若发送端在规定时间内未收到来自接收端的确认信息（即计时器超时），则重发存放于缓冲器中的待确认数据帧（如图 6-16 右图所示的 data1）。

（4）后续数据帧的发送步骤与 data0 帧的发送一样。

现在来打个与这里所介绍的“空闲重发请求”差错控制方案类似的比方。一个牙牙学语的小孩，你要他从 1 数到 10。由于他非常小，对这十个数字记得并不牢，所以他会一个个地去数，而且每数完一个数，他都会停下来，抬起头望着你，等待你的点头或者说“对”，然后他才继续往下数。你对他点头或者说“对”就相当于你向他发回了一个确认消息。

从以上过程可以看出，“空闲重发请求”方案除了要求在发送的数据帧中携带一定的校验码外，还要求发送端和接收端都有一个用于存放待确认的发送帧和待向“网络层”提交的数据帧的缓冲存储空间，但这个存储空间要求比较小，因为它只需要临时存放一个数据帧。

由以上分析可以看出，这种 ARQ 方法设计简单，容易实现，但是这种方法也有一个不可克服的缺点，那就是每传送一个数据帧都要有一个等待时间（称为“占空时间”），信道的有效利用率低。“占空时间”与传送一个帧的全部时间的比例称为“占空比”。数据帧越短，“占空比”就越大，相当于信道的利用率越低；数据帧越长，“占空比”就越小，信道的利用率就越高。但是数据帧越长，出错的概率也越大，从而出现多次重发，因此传输效率也不会太高。正因为如此才有下面将要介绍的“连续重发请求”差错控制方案。

6.3.5 连续重发请求的回退 N 帧策略

为了减小占空比，提高传输效率，人们又提出了“连续重发请求”（Continuous ARQ）的差错控制方案。

“连续重发请求”方案是指发送端可以连续发送一系列数据帧（也不是总是不断地发送，具体可以连续发送多少个帧要视双方的缓存空间大小（亦即“窗口大小”）而定），即不用等前一帧被确认便可继续发送下一帧，效率大大提高。当然，在这个连续发送的过程中也可以接收来自接收端的响应消息（可以是确认帧，也可以是否认帧），发送端同样可以对传输出错的数据帧（如接收端返回了否认帧的帧或者响应计时器超时的帧）进行重发，具体处理方法在本节后面介绍。

由于“连续重发请求”方案减少了等待时间，整个通信的吞吐量就提高了，但在这种重发请求方案中，需要在发送端设置一个较大的缓冲存储空间（称为重发表），用以存放若干待确认的数据帧。当发送端收到对某数据帧的确认帧后，便可从重发表中将该数据帧删除。

当出现传输差错时，“连续重发请求”方案有两种处理策略，即“回退 N 帧”（Go-back-N）策略和“选择重发”（Selective Repeat）策略。

“回退 N 帧策略”的基本原理是，如果发送端一共发送了 n 个数据帧（编号从 0 一直到 n-1），但收到接收端发来的 ACK 确认帧中少了某一个或几个帧的 ACK 确认帧（要么是数据帧出现了丢失，要么是 ACK 帧或 NAK 帧出现了丢失，最终造成 ACK 确认帧序号不连续），或者在接收某一帧时检测出有错，接收端发送一个 NAK 否认帧给发送端，发送端则可以判断接收端最后一个正确

接收的帧编号，然后从缓存空间的重发表中重发所收到的最后一个 ACK 帧序号以后的所有帧。

如图 6-17 所示是一个“回退 N 帧”差错控制的示例。图中假定发送完 8 号帧后，发 0 号和 1 号帧的 ACK 帧已收到，但现 2 号帧的 ACK 帧在计时器超时后还未收到，则发送端只能退回到从 2 号帧开始重发以后所有已发的数据帧（2～8 号共 7 个帧），尽管或许后面 3，4，5，…，8 号帧的 ACK 确认帧已收到。当然原来已发并且接收端已正确接收的这些帧接收端会自动删除的，采用最新接收到的同序号帧。

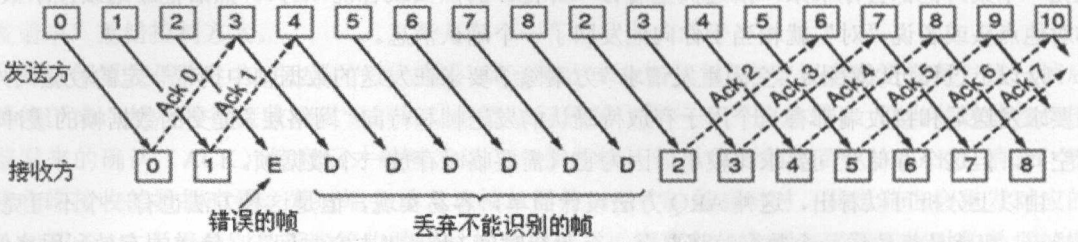


图 6-17 “回退 N 帧”策略差错控制原理示意图

下面我们将逐层深入地分析“回退 N 帧”策略中的数据处理流程。

1. 理想情形下的数据处理流程

先来看一下理想状态下的情形，也就是数据帧和确认帧都不发生差错或丢失的情形（也就是不存在有 NAK 帧返回，不存在数据帧和 ACK 帧丢失，也不存在计时器超时）下的数据处理流程：

（1）发送端连续发送数据帧，而不等待任何数据帧的 ACK 帧的返回。

（2）发送端在重发表中保存所发送的每个数据帧的备份。

（3）接收端对每一个正确收到的数据帧返回一个 ACK 帧，ACK 帧中包括对应帧的编号。

（4）接收端保存一个接收（次序）表，包含最后正确收到的数据帧的编号。

（5）当发送端收到相应数据帧的 ACK 帧时，发送端即从重发表中删除该数据帧。

2. 存在帧差错情形下的数据处理流程

接着考虑帧（包括数据帧和响应帧）出现差错的情形下“回退 N 帧”策略的数据处理流程，这里的“差错”既包括数据帧在接收端检测出差错，也包括数据帧或响应帧在传输过程中出现丢失。此时，“回退 N 帧”策略中的数据处理流程如下：

（1）假设发送的第(N+1)个帧发生了差错，接收端要么检测出第(N+1)帧有错，要么发现没有接收到(N+1)帧，反而接收到了第(N+2)帧或第(N+3)帧等。

（2）出现这种情况时，接收端立即返回一个相应的未正确接收的否定帧 NAK(N+1)，预示接收端最后正确收到的是第(N)帧（N+1 帧的前一帧）；同时对后面每个失序的数据帧，接收端都产生相应的 NAK 帧，否则如果所发送的 NAK(N+1)帧正好丢失或出错，将产生死锁，即发送端不停地发送新的帧，同时等待对第(N+1)帧的确认，而接收端不停地清除后继的帧，当然可以通过超时机制或者流量控制来避免死锁的发生（如“滑动窗口法”，具体将在本章后面介绍“数据链路层”的“流量控制”原理时介绍）。

（3）发送端在收到NAK(N+1)帧或者收到了NAK(N+2)、NAK(N+3)等帧时，从重发表中重发第(N+1)帧或者对应的NAK帧中序号所对应的帧，同时接收端清除所有失序的帧（从第(N+2)帧或者对应NAK帧中序号所对应的帧开始，直到重新正确接收到重发的第(N+1)帧或者对应NAK帧中序号所对应的帧）。

（4）接收端重新收到第(N+1)帧或者对应的NAK帧中序号所对应的帧，接收端就继续正常操作。

以上流程看似比较复杂，其实原理很简单。也来打个比方吧，比如我们经常要自己的小孩从1数到100，以检验他的数数能力。小孩数着数着，突然你发现某个数数得不正确（如数到42时本应是43，却数成了53）或者中间漏了一个或多个数（如数到42时，本应是43却数成跳过了，从45开始数了），你就会要求小孩重新从出现错误的那个数开始重数（如从43开始数），尽管在后面大多数是数得正确的。这就是“回退N帧”策略的差错控制原理。

6.3.6 连续重发请求的选择重发策略

通过上节的学习可以看出，“回退N帧”策略因为可以连续发送数据帧而提高了传输效率，但也有不利的方面，那就是在重发时必须把原来已正确传送过的数据帧再次发送，仅仅是因为这些数据帧之前的某个数据帧或确认帧发生了差错，这样又使传输效率降低了。所以当通信链路的传输质量很差、误码率较大时，“回退N帧”策略就没有什么优势了，因为这时可能经常要重传大量的数据帧。

为了弥补“回退N帧”策略的不足，另一种效率更高的差错控制策略——“选择重发”策略诞生了。在这个差错控制策略中，当接收端发现某帧出错后，其后继续送来的正确帧虽然不能立即递交给接收端的“网络层”，但接收端仍可接收下来，先存放在一个缓冲区中，同时通过向发送端发送NAK否认帧要求发送端重新传送出错的那一帧。一旦收到重新发来的正确帧后，就可以与原已存于缓冲区中的其余帧一起按正确的顺序递交给“网络层”。

“选择重发”策略规定，当发送端收到包含出错帧序号的NAK帧时，据此序号从重发表中选出相应帧的备份，直接插入到发送帧队列的前面给予重发，因为重发表的帧重发是按照FIFO（先进先出）的机制进行排列的，插在前面是为了可以最先重发，避免了对后继正确数据帧的多余重发，使传输效率明显提高了。

如果仍用小孩数数来打比方的话，就是你先让小孩自己一直数下去，发现数错时你记下来（不要打断他正常的数数流程），当小孩数完后你再根据这些错误要求小孩重数对应的数就行了，而不必要求他从错误的地方重新数一遍。

下面也分两种情况来讨论“选择重发”策略的数据处理方法。

1. 数据帧出现差错情形下的处理流程

当数据帧有差错（包括接收端检测到所接收的数据帧有差错和有数据帧丢失两种可能）时，“选择重发”策略的数据处理流程如下：

（1）发送端连续发送多个数据帧，接收端对每个已正确接收的数据帧返回一个 ACK 帧；现假设第(N+1)个帧出现差错或丢失，如果检测到第(N+1)个帧有错误，则向发送端返回一个否定 NAK(N+1)帧；如果一直在收到第(N+2)全数据帧时还没有收到第(N+1)帧，则表明该帧已丢失，接收端不产生任何动作；但无论结果如何，仍会对所有已正确接收的数据帧，如第(N)个帧、第(N+2)个帧、第(N+3)个帧等向发送方返回确认 ACK 帧。

（2）当发送端收到来自接收端的否定 NAK(N+1)或者收到第(N+2)帧的 ACK 帧时，则会检测出其失序（因为按顺序，在收到 ACK(N+2)帧之前应该是收到 ACK(N+1)帧），得知第(N+1)帧没有被确认。将第(N+2)帧从重发表中清除，并在继续发送后继数据帧之前重发第(N+1)帧。

图 6-18 所示为由于接收端检测到 2 号帧有错，向发送端发送了一个否认帧 NAK2，要求发送端选择重发 2 号帧的示意图。从中可以看出，“选择重发”策略下只需发送有错的帧，而不会发送从有错帧开始后面所有的帧，显然减少了信道资源浪费，提高了传输效率，但要求发送端和接收端都有足够大的缓冲区空间，以便存储多个帧的重发表和预提交数据帧。

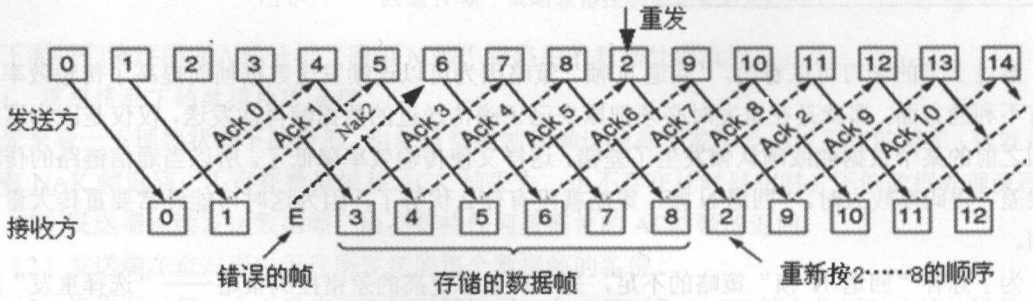


图 6-18 “选择重发”策略差错控制原理示意图

2. 响应帧出现差错情形下的数据处理流程

在响应帧（包括确认 ACK 帧和否认 NAK 帧）出现差错时，也就是本应接收的是第(N)个帧的响应帧而接收到了第(N+1)个帧的响应帧情况下的数据处理流程如下（现以 ACK 帧出错进行介绍，与 NAK 帧出错的处理流程一样）：

（1）发送端已收到了第(N-1)个帧的 ACK 帧，接下来应该收到的是第(N)个帧的 ACK 帧，而偏偏收到的是第(N+1)个帧的 ACK 帧。

（2）发送端收到了 ACK(N+1)帧，检测出在重发表中第(N)个帧都还没收到 ACK 帧，因此认为第(N)个帧出现了差错(事实上并不是这样的)，重发第(N)个帧。

（3）接收端收到发送端重发的第(N)个帧，则搜索接收表并确定第(N)个帧已被正确接收，因此认定这个重发的第(N)个帧是一个重份，于是直接删除这个重发的第(N)个帧并返回一个 ACK(N)给发送端，以使发送端从重发表中删除将第(N)个帧。这样就达到了响应帧出现差错时的错误纠正。

6.4 海明纠错码

海明码（Hamming Code）是一个可以有多个校验位，具有检测并纠正一位错误代码的纠错码，所以它也仅用于信道特性比较好的环境中，如以太局域网中，因为信道特性不好的情况下，出现的错误通常不是一位。

海明码的检错、纠错基本思想如下：

（1）将有效信息按某种规律分成若干组，每组安排一个校验位通过异或运算（结果为 0）得出具体的校验码。

（2）在接收端同样通过异或运算看各组校验结果是否正确（结果为 0 时正确）。出错时观察出错的校验组的校验位或者多个出错的校验组的共同校验位，得出具体的出错比特位。

（3）对错误位取反来将其纠正，亦即原来是 1 就变成 0，原来是 0 就变成 1。

要采用海明码纠错，需要按以下步骤来进行：计算校验位数→确定校验码位置→确定校验码→实现校验和纠错。为了方便阅读，下面分几个小节来具体介绍这几个步骤。

6.4.1 计算校验位数

要使用海明码纠错，首先就要确定发送的数据所需要的校验码（也就是“海明码”）位数（也称“校验码长度”）。它是这样规定的：假设用 N 表示添加了校验码位后整个信息的二进制位数，用 K 代表其中有效信息位数，r 表示添加的校验码位，它们之间的关系应满足：$N=K+r \leqslant 2^r-1$。

这个关系是为了确保 r 位校验码能校验全部的数据位，因为 r 位校验码所能表示的最大十进制数为 2^r-1，同时也确保各位校验码本身不被其他校验码校验。

如 K=5，则要求 $2^r-r \geqslant 5+1=6$，根据计算可以得知 r 的最小值为 4，也就是要校验 5 位信息码，则要插入 4 位校验码。如果信息码是 8 位，则要求 $2^r-r \geqslant 8+1=9$，根据计算可以得知 r 的最小值也为 4。根据经验总结，得出信息码位数和校验码位数之间的关系如表 6-1 所示。

表 6-1 信息码位数与校验码位数之间的关系

信息码位数	1	2～4	5～11	12～26	27～57	58～120	121～247
校验码位数	2	3	4	5	6	7	8

6.4.2 确定校验码位置

通过上节已确定了对应信息中要插入的校验码位数，但这还不够，因为这些校验码不是直接附加在信息码的前面、后面或中间的，而是分开插入到不同的位置。但不用担心，校验码的位置是很容易确定的，那就是校验码必须是在 2^n 次方位置（n 从 0 开始），如第 1，2，4，8，16，32，…位（对应 2^0，2^1，2^2，2^3，2^4，2^5，…，是从最左边的位数起的），这样一来就知道了信息码的分布位置，也就是非 2^n 次方位置，如第 3，5，6，7，9，10，11，12，13，…位（也是从最左边的

位数起的）。

举一个例子，假设现有一个 8 位信息码，即 b1b2b3b4b5b6b7b8，由表 6-1 得知，它需要插入 4 位校验码，即 p1、p2、p3、p4，也就是整个经过编码后的数据码（称为“码字”）共有 12 位。根据以上介绍的校验码位置分布规则可以得出，这 12 位编码后的数据就是 p1p2b1p3b2b3b4p4b5b6b7b8。

现假设原来的 8 位信息码为 10011101，因现在还没有求出各位校验码值，现在这些校验码位都用“？”表示，最终的码字为：？？1？001？1101。

6.4.3 确定校验码

经过前面两节我们已经确定了所需的校验码位数和这些校验码的插入位置，但这还不够，还得确定各个校验码值。这些校验码的值不是随意的，每个校验位的值代表了代码字中部分数据位的奇偶性（最终要根据是采用奇校验还是偶校验来确定），其所在位置决定了要校验的比特位序列。

总的原则是：第 i 位校验码从当前位开始，每次连续校验 i 位（这里是数值 i，不是第 i 位，下同）后再跳过 i 位，然后再连续校验 i 位，再跳过 i 位，依此类推。最后根据所采用的是奇校验还是偶校验即可得出第 i 位校验码的值。

1. 计算方法

下面是校验码的具体计算方法。

p1（第 1 个校验位，也是整个码字的第 1 位）的校验规则是：从当前位数起，校验 1 位，然后跳过 1 位，再校验 1 位，再跳过 1 位，……。这样就可以得出 p1 校验码位可以校验的码字位包括：第 1 位（也就是 p1 本身，等于 2^{1-1}）、第 3 位、第 5 位、第 7 位、第 9 位、第 11 位、第 13 位、第 15 位，……。然后根据所采用的是奇校验还是偶校验最终可以确定该校验位的值。

p2（第 2 个校验位，也是整个码字的第 2 位）的校验规则是：从当前位数起，连续校验 2 位，然后跳过 2 位，再连续校验 2 位，再跳过 2 位，……。这样就可以得出 p2 校验码位可以校验的码字位包括：第 2 位（也就是 p2 本身，等于 2^{2-1}）、第 3 位，第 6 位、第 7 位，第 10 位、第 11 位，第 14 位、第 15 位，……。同样根据所采用的是奇校验还是偶校验最终可以确定该校验位的值。

p3（第 3 个校验位，也是整个码字的第 4 位）的校验规则是：从当前位数起，连续校验 4 位，然后跳过 4 位，再连续校验 4 位，再跳过 4 位，……。这样就可以得出 p4 校验码位可以校验的码字位包括：第 4 位（也就是 p3 本身，等于 2^{3-1}）、第 5 位、第 6 位、第 7 位，第 12 位、第 13 位、第 14 位、第 15 位，第 20 位、第 21 位、第 22 位、第 23 位，……。同样根据所采用的是奇校验还是偶校验最终可以确定该校验位的值。

p4（第 4 个校验位，也是整个码字的第 8 位）的校验规则是：从当前位数起，连续校验 8 位，然后跳过 8 位，再连续校验 8 位，再跳过 8 位，……。这样就可以得出 p4 校验码位可以校验的码字位包括：第 8 位（也就是 p4 本身，等于 2^{4-1}）、第 9 位、第 10 位、第 11 位、第 12 位、第 13 位、第 14 位、第 15 位，第 24 位、第 25 位、第 26 位、第 27 位、第 28 位、第 29 位、第 30 位、第 31 位，……。同样根据所采用的是奇校验还是偶校验最终可以确定该校验位的值。

……

最终得出如下规律：pn 校验码位校验的码字位为：第 2^{n-1} 位（也就是 pn 本身）、第 $2^{n-1}+1$ 位、第 $2^{n-1}+2$ 位、第 $2^{n-1}+3$ 位、……、第 2^n-1 位，第 $3\times2^{n-1}$ 位、第 $3\times2^{n-1}+1$、……、第 $2\times2^n-1$ 位，第 $5\times2^{n-1}$ 位、第 $5\times2^{n-1}+1$ 位、第 $3\times2^n-1$ 位，……，第 $7\times2^{n-1}$ 位、第 $7\times2^{n-1}+1$ 位、……、第 $7\times2^n-1$ 位，……，第 $(2m-1)\ 2^{n-1}$ 位、……第 $m\times2^n-1$ 位。图 6-19 列出了前 5 个校验码所校验的码字位，bn 为数据位。

码字中的位	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
对应的位	p1	p2	b1	p3	b2	b3	b4	p4	b5	b6	b7	b8	b9	b10	b11	p5	b12	b13
p1校验的位	√		√		√		√		√		√		√		√		√	
p2校验的位		√	√			√	√			√	√			√	√			√
p3校验的位				√	√	√	√					√	√	√	√			
p4校验的位								√	√	√	√	√	√	√	√			
p5校验的位																√	√	√

图 6-19　前 5 个校验码所校验的码字位

最后每组通过异或逻辑运算（与偶校验原理一样）使每组的运算结果为 0，即可得出第 i 位校验码的值。

2. 校验码计算示例

同样举上面的例子来说明，码字为**? ? 1? 001? 1101**。

先求第 1 个“？”（也就是 p1，第 1 位）的值，因为整个码字长度为 12（包括信息码长和校验码长），所以可以得出本示例中 p1 校验码校验的位数是 1、3、5、7、9、11 共 6 位。这 6 位中除了第 1 位（也就是 p1 位）不能确定外，其余 5 位的值都是已知的，分别为 1、0、1、1、0。现假设采用的是偶校验（也就是要求整个被校验的位中的“1”的个数为偶数），从已知的 5 位码值可知，已有 3 个“1”，所以此时 p1 位校验码的值必须为“1”，得出 p1=1。

再求第 2 个“？”（也就是 p2，第 2 位）的值，根据以上规则可以很快得出本示例中 p2 校验码校验的位数是 2、3、6、7、10、11，也是一共 6 位。这 6 位中除了第 2 位（也就是 p2 位）不能确定外，其余 5 位的值都是已知的，分别为 1、0、1、1、0。现假设采用的是偶校验，从已知的 5 位码值可知，也已有 3 个“1”，所以此时 p2 位校验码的值必须为“1”，得出 p2=1。

再求第 3 个“？”（也就是 p3，第 4 位）的值，根据以上规则可以很快得出本示例中 p3 校验码校验的位数是 4、5、6、7、12，一共 5 位。这 5 位中除了第 4 位（也就是 p3 位）不能确定外，其余 4 位的值都是已知的，分别为 0、0、1、1。现假设采用的是偶校验，从已知的 4 位码值可知，也已有 2 个“1”，所以此时 p3 位校验码的值必须为“0”，得出 p3=0。

最后求第 4 个“？”（也就是 p4，第 8 位）的值，根据以上规则可以很快得出本示例中 p4 校验码校验的位数是 8、9、10、11、12（本来是可以连续校验 8 位的，但本示例的码字后面的长度没有这么多位，所以只校验到第 12 位止），也是一共 5 位。这 5 位中除了第 8 位（也就是 p4 位）不能确定外，其余 4 位的值都是已知的，分别为 1、1、0、1。现假设采用的是偶校验，从已知的 4 位码值可知，已有 3 个“1”，所以此时 p4 位校验码的值必须为“1”，得出 p4=1。

最后就可以得出整个码字的各个二进制值码字为 **11100011101**（带阴影的 4 位就是 CRC 校验码）。

6.4.4 实现校验和纠错

虽然前面已经把各位校验码计算出来了，但是如何检测出哪一位在传输过程中出了差错呢？它又是如何实现对错误的位进行纠正的呢？本节就来具体分析。

1. *海明码的差错检测*

海明码校验的方式就是各校验码与它所校验的各位信息码进行“异或运算”，即：

G1=p1⊕b1⊕b2⊕b4⊕b5⊕……

G2=p2⊕b1⊕b3⊕b4⊕b6⊕b7⊕b10⊕b11⊕……

G3= p3⊕b2⊕b3⊕b4⊕b8⊕b9⊕b10⊕b11⊕……

G4= p4⊕b5⊕b6⊕b7⊕b8⊕b9⊕b10⊕b11⊕……

G5= p5⊕b12⊕b13⊕b14⊕b15⊕b16⊕b17⊕b18⊕b19⊕b20⊕b21⊕b11⊕b23⊕b24⊕b25⊕b26⊕……

以上 G1，G2，…代表对应校验组。正常情况下（也就是整个码字不发生差错的情况下），在采用偶校验时，各校验组通过异或运算后的校验结果均应该为 **0**，也就是前面所说的 G1，G2，G3，G4，…均为 0，因为此时 1 为偶数个，进行异或运算后就是 0。

如果数据在传输途中出了差错，要判断出具体的出错位，就需要理解海明码的检错原理。首先，海明码是一个多重校验码，而且每个信息码位会同时被多个校验码（至少 2 个，参见图 6-15）进行校验；另外，每个校验码只能被自己校验。通过这样的关系可以得出海明码在检错方面的以下两个规律：

（1）所有校验码所在的位只由对应的校验码进行校验。

如果校验码在传输过程中出现了差错，影响的只是对应的校验组的校验结果，不会影响其他校验组的校验结果。这点很重要，如果最终发现只是一个校验组中的校验结果不符，则直接可以知道是对应校验组中的校验码在传输过程中出现了差错。

（2）所有信息码位均被至少两个校验码进行了校验，也就是至少校验了两次。

如果只是信息码出了错误，则至少有两组校验的结果出错，不可能只有一组校验结果出错。查看是哪些校验组包括了相同的码字位就可以很快确定是哪位信息码在传输过程中出了差错。

现假设经过海明码编码后的二进制字符串为 **11100011101**（一共 12 位，带阴影的 4 位就是校验码），到了接收端进行海明码校验时发现只有 G4=1（也就是只有这组校验结果不等于 0），判断到底是哪位在传输过程中出了差错。

通过前面介绍的校验规律可以很快地发现是 G4 校验组中的 p4 校位码（也就是整个码字中的第 8 位）错了，因为只有一组校验结果出现差错时，则肯定只是对应的校验位出了差错，也就是最终的码字变成了 11100000**0**1101。

如果 G3、G4 两个校验值都不为 0，也就是都等于 1，则通过图 6-15 比较 G3、G4 两个校验组

（注意本示例中码字长度一共才 12 位，只需要比较前 12 位）中共同校验的码位可以很快发现是 b8，也就是第 12 位出现了差错，也就是最终的码字变成了 111000011100。



在这里一定要注意，最终有多少个校验组出现差错也不是随意的，一定要结合实际传输的码字长度来考虑。如上例是一共 12 位，如果换成了是 16 位的码字，且当 b9 位出现差错时，则 G1、G3、G4 一定会同时出现错误，因为 b9 这个位是 3 个校验组同时校验的，只要它一出错，肯定会同时影响这 3 个校验组的值。同理，如果是 b11 位出现了差错，因为它同时受 G1、G2、G3、G4 四个校验组校验，所以这 4 个校验组结果都将出现错误。

2. 海明码的差错纠正

检测出了是哪位差错还不够，因为海明码具有纠正一位错误的能力，所以还需要完成纠错过程。这个过程的原理比较简单，就是直接对错误的位进行取反或者加“1”操作，使它的值由原来的“1”变成“0”，由原来的“0”变成“1”（因为二进制中每一位只能是这二者之一）。

以上就是海明码的差错检测和差错纠正原理了，确实比单纯的奇偶校验码复杂一些，但只要理清了思路，也还是比较简单的。

6.5 流量控制

其实在上节介绍“差错控制”功能时就提到了“流量控制”功能，因为一些差错控制方案本身就具有一定的流量控制功能，如前面介绍的“空闲重发请求”方案中规定，发送端每发完一个数据帧后把这个帧保存在缓存空间中，然后就停下来等待接收端发来的确认消息，收到确认信息后才能继续发送下一帧，这就可以控制链路中的数据流量。在“连续重发请求”方案中，虽然发送端可以一次发送多个数据帧，但是也不是没有限制的，因为发送端需要把每次发送的数据帧保存在缓存空间中，接收端也要把向“网络层”提交的数据帧先保存在缓存空间中，所以最终一次能发送多少个帧是由双方缓存空间大小决定的。这就是本节后面将要提到的“窗口大小”。

数据链路层的流量控制方案主要有两种：一种是适用于面向字符的异步通信协议（如 RS-232）中的简单流量控制方案——XON/XOFF（继续/停止）方案；另一种是适用于大量数据通信环境中的“滑动窗口机制”。

6.5.1 XON/XOFF 流量控制方案

XON/XOFF（Transmitter on/Transmitter off，继续传输/停止传输）是一种流量控制协议，常用于在数据传输速率大于等于 1200b/s，而接收端数据处理速率远小于这个值时（也就是通信双方速率不同步），通过对发送端的数据传输速率进行控制，以达到与接收数据数据处理速率匹配。XON/XOFF（继续/停止）是一种最简单的流量控制技术，主要适用于异步通信中，通过接收端使

用特殊字符来控制发送端数据的发送。

XON/XOFF 流量控制方案的基本思想是：当接收端认为不能继续接收数据时（也就是接收端的缓存空间满了时或者接近满了时），接收端会向发送端发送一个 XOFF 控制字符，当发送端收到对应的 XOFF 控制字符时就停止数据的继续发送；当接收端认为可以继续接收数据时，接收端会再向发送端发送一个 XON 控制字符，发送端收到这个控制字符后就知道可以恢复数据发送了，继续发送数据，一直这么循环下去。

其中 XON 采用 ASCII 字符集中的控制字符 DC1（十进制值为 17，十六进制值为 11，相当于按下 Ctrl+Q 组合键），XOFF 采用 ASCII 字符集中的控制字符 DC3（十进制值为 19，十六进制值为 13，相当于按下 Ctrl+S 组合键）。在一次数据传输过程中，XOFF、XON 的周期可重复多次，但这些操作对用户来说是透明的，也就是说用户不用管它，设备会自动操作。

许多异步数据通信软件均支持 XON/XOFF 协议。这种方案也可用于计算机向打印机或其他终端设备（如 Modem 的串行通信）发送字符，在这种情况下，打印机或终端设备中的控制部件用以控制字符流量。如我们在通过 Modem 拨号连接网络时就是采用这种流量控制方法。当从 PC 上的数据到达 Modem 时，如果 Modem 中的缓存空间满了，它就会向 PC 发送一个代表“停止传输”的 XOFF 控制字符，而当 Modem 中的缓存空间没满时，Modem 又会向 PC 传送一个代表“继续传输”的 XON 控制字符。

再如，在局域网中一台 PC 机连接了一台打印速度比较慢的打印机，当 PC 机开始向打印机发送要打印的文件时，因为 PC 机的数据传输速率非常高，有许多文件打印机根本很难及时打印。此时打印机会向 PC 机发送一个 XOFF 字符来通知 PC 机，要求暂停文件的发送。PC 机上的软件看到来自打印机的 XOFF 控制字符后，就会暂停文件的发送；而当打印机中排队等候打印的文件打完了或者打印得差不多了时，打印机又会向 PC 机发送一个 XON 控制字符，通过 PC 机可以继续发送要打印的文件。

6.5.2 滑动窗口机制

在上面介绍的 XON/XOFF 流量控制方案中，为了实现发送端与接收端的速率匹配，需要往返发送一些特殊的控制字符，这样就会使信道的利用率大打折扣，主要是用于与一些低数据处理能力的接收端通信时采用。在实际的数据链路层流量控制中，更多的是采用本节将要介绍的“滑动窗口机制”来进行流量控制（传输层也有这样的流量控制方案，具体将在第 10 章中介绍）。

1. 理解“滑动窗口”机制

“滑动窗口机制”中的“窗口”是指发送端和接收端的缓存空间大小；“滑动”的意思是指这个窗口大小不是固定的，是变化的，因为缓存空间中存放的未处理帧数是变化的，发送端在收到确认帧后会删除原来保存在缓存中的待重发帧，这样可用于数据发送的缓存“窗口”就是变化的；同时接收端向“网络层”提交了一个帧后也会删除原来保存在缓存中的帧，这样可用于接收数据的缓存“窗口”也是变化的。

至于“缓存空间大小”采取不同的流量控制方案有不同的值，但要明白的是，缓存空间都是非

常有限的，就像计算机 CPU 中的缓存一样。缓存越大，成本越高。如在 6.3.4 节介绍的“空闲重发请求”方案中一次只能发送一个帧，发完一个帧后就等待来自接收端的确认帧，收到确认帧后就删除原来保存在缓存空间中的待重发帧，接收端不需要缓存空间，因为它接收到数据后即可进行处理，对于有错误的帧直接丢弃。所以在“空闲重发请求”方案中仅发送端需要可以保存一个帧的缓存空间，也就是它的“缓存空间大小”就是一个帧。

在 6.3.5 节介绍的“连续重发请求”方案中，发送端一次可以连续发送多个帧，并且在其缓存空间中保存所有已发但没有接收到来自接收端确认帧的待重发帧，而不用像“空闲重发请求”那样发一帧停下来等待接收端的确认帧，接收端也可以在缓存空间中保存来不及处理的帧，大大提高了数据传输的效率。

同样打个简单的比喻来说，就像一个水缸接了大小两个不同口径的水管，进水管的口径较大，出水管的口径较小。在这样的情况下，进水管肯定不能持续不断地向水缸中加水，否则就会因为出水速率不匹配导致水从水缸中溢出来。这时就要求进水管供了一段时间的水后要停下来，等水缸中的水用得差不多了再加水。这时水缸的容量就相当于上面所说的接收端“缓存空间”大小。

这里涉及到一个非常重要的问题，那就是到底一次最多可以连续发多少个帧比较合适呢？这里要考虑两方面的因素：一是要能实现有效的差错控制，二是要与接收端的数据处理能力相匹配，前者我们已在 6.3.5 节中介绍，本节仅从后者来考虑，也就是从流量控制角度来考虑。

另外，缓存空间大小又与帧编号所用位数有关，因为在数据传输时每个帧都是有序列号的，这在本章前面介绍的“差错控制”方案中就已经说到。缓存空间越大，用于表示帧编号的位数就要越多，如用 1 位表示帧编号的话，可以表示 2 个帧（也就是“窗口大小”为 2），即 0 号帧和 1 号帧；用 2 位表示帧编号的话，可以表示 4 个帧（也就是“窗口大小”为 4），即 0、1、2、3 号帧；用 n 位表示帧编号的话，可以表示的帧数是 2^{n-1}。

用于表示帧编号的位数越多，帧的无用开销就越大，因为一般情况下对于同种数据链路层协议的帧，它的总位数是固定的。但是用于表示帧编号的位数太少，在帧传输过程中很容易出现混乱，因为到时保存在缓存中的帧可能存在多个相同但实际内容不同的帧。为了权衡两方面的关系，像在 SDLC 和 HDLC 这样的帧中（以太网、PPP 等协议帧是不带帧编号的，不采用这种流量控制方式），帧编号一般是采用 3 位的。

2. *滑动窗口实例*

为了方便解释，在此仅以 1 位帧序列号，也就是窗口大小为 2 的示例来介绍“滑动窗口机制”。下面从初始状态开始介绍整个流程（注意，这里仅考虑正常传输的情况，不考虑出现差错的情况），如图 6-20 所示（各步对应图中的相应序号）：

（1）在初始状态下，发送端和接收端的缓存空间中均没有保存数据帧。

（2）发送端开始发送第一个帧——0 号帧，把它保存在缓存空间中并建立一个待重发表，第一个帧号就是“0”。

（3）因为是一个比特位用于帧编号，窗口大小为 2，所以发送端还可以继续发送第二个帧——1 号帧。同时也把这个帧保存在缓存空间中，并向待重发表中添加第二个帧的序号“1”。此时

因为在发送端的缓存空间中已保存了两个帧（0 号帧和 1 号帧），达到了窗口大小的值，不能继续发送后面的帧了，先停下来等待来自接收端的确认帧。在等待的过程中，接收端可能收到了 0 号帧，然后向发送端返回 0 号帧的确认消息。

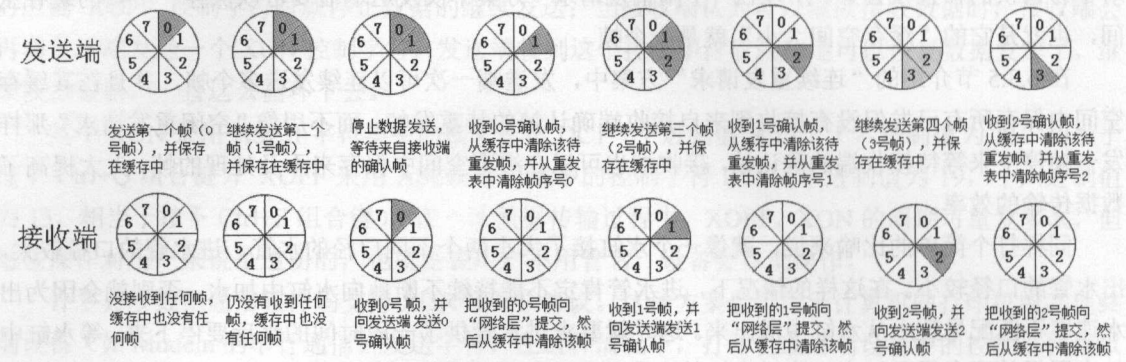


图 6-20　“滑动窗口机制”工作流程示例

（4）发送端在收到 0 号帧的确认帧后，立即从缓存空间中清除原来保存的 0 号待重发帧，并在待重发表中清除帧序号“0”；接收端把收到的 0 号帧向“网络层”提交，清除缓存空间中的 0 号帧，此时缓存空间中又为空了。

（5）此时发送端继续发送第三个帧——2 号帧（实际帧编号又是 0），同时也把这个帧保存在缓存空间中，并向待重发表中添加第三个帧的序号“2”。因为此时发送端的缓存空间中又已保存了两个帧（1 号帧和 2 号帧），又达到了窗口大小的值，不能继续发送后面的帧了，要先停下来等待来自接收端的确认帧。在等待的过程中，接收端可能又收到了 1 号帧，然后向发送端返回 1 号帧的确认帧。

（6）发送端在收到 1 号帧的确认帧后，立即从缓存空间中清除原来保存的 1 号待重发帧，并在待重发表中清除帧序号“1”。接收端把收到的 1 号帧向“网络层”提交，清除缓存空间中的 1 号帧，此时缓存空间中又为空了。

（7）此时发送端继续发送第四个帧——3 号帧（实际帧编号又是 1），同时也把这个帧保存在缓存空间中，并向待重发表中添加第四个帧的序号“3”。同样，因为此时发送端的缓存空间中又已保存了两个帧（1 号帧和 2 号帧），又达到了窗口大小的值，不能继续发送后面的帧了，要先停下来等待来自接收端的确认帧。在等待的过程中，接收端可能又收到了 2 号帧，然后向发送端返回 2 号帧的确认帧。

（8）发送端在收到 2 号帧的确认帧后，立即从缓存空间中清除原来保存的 2 号待重发帧，并在待重发表中清除帧序号“2”。接收端把收到的 2 号帧向“网络层”提交，清除缓存空间中的 2 号帧，此时缓存空间中又为空了。

……

后面的步骤就是按照以上流程类推了。

6.6 面向字符的 BSC 同步传输协议

在第 4 章中介绍了两种数据传输模式：同步传输和异步传输，在同步传输中需要确保在接收端能明确区分传输单位之间的边界，即同步问题。本节和后面几节将分别介绍在广域网中几种典型链路层协议的同步方法。

在广域网中，数据链路层中常见的协议可分为两大类："面向字符"的链路层协议和"面向比特"的链路层协议（这些都属于同步传输模式协议）。

"面向字符"的同步方法也称"字符填充的首尾定界符法"。在该同步方法中，数据帧中的数据都被看做字符序列（所以称之为"面向字符"的同步传输），所有的控制信息也都是字符形式（当然数据的表示形式还是二进制的比特流），每个数据块的头部用一个或多个同步字符 SYN 来标记数据块的开始，尾部用字符 ETX 来标记数据块的结束。

"面向字符"的链路层协议主要有：IBM BSC（Binary Synchronous Communication，二进制同步通信）协议、DEC DDCMP（Digital Data Communications Message Protocol，数字数据通信消息协议）、SLIP（Serial Line Internet Protocol，串行线路网际协议）、PPP（Point-to-Point Protocol，点对点协议）等。本节先来介绍 BSC 协议的同步传输方法。

6.6.1 BSC 控制字符和数据块结构

面向字符的同步传输协议的典型代表就是 IBM 公司的 BSC 协议。BSC 协议规定，链路上传送的数据必须是由规定字符集（可以是 ASCII 或 EBCDIC（Extended Binary Coded Decimal Interchange Code，扩展二进制－十进制交换码））中的字符组成，控制信息也必须由同一个字符集中的若干指定的控制字符构成。

BSC 协议与所有面向字符的同步传输协议一样，也是一次可以传送由若干个字符组成的数据块（通常是一帧），而不是一次只传送一个字符。同时规定了十种特殊字符（称为"通信控制字符"）作为这个数据块的开始与结束标志以及整个传输过程的各种控制信息标志（并不是每个数据块中都有这十种全部的控制字符）。这十种通信控制字符说明如下：

- ACK（Acknowledge）：确认标志，是由接收端发出的，作为对正确接收到报文的响应。
- DLE（Data Link Escape）：转义标志，用以指示后面的字符是数据字符，而不是特殊控制字符。这是用来进行透明传输的，使得在报文中也存在这十个控制字符时，在这些字符前加上 DLE 字符后，通知接收端把它们当作普通的报文处理，而不是作为控制字符来识别。具体将在本节后面介绍。
- ENQ（Enquire）：询问标志，用以请求远程站点给出响应。响应可能包括远程站点的身份或状态。
- EOT（End of Transmission）：发送完毕标志，用以表示一个或多个文本的发送结束，并拆除链路。

- ETB（End of Transmission Block）：块终止或组终止标志，用以标识每个数据块的结束位置。仅在一个报文要分成多个数据块传输时才有此标志。
- ETX（End of Text）：文本终止标志，标识报文文本的结束。仅在一个报文不分成多个数据块传输时才有此标志。
- NAK（Negative Acknowledge）：否认标志，是由接收端发出的，作为对未正确接收的报文响应。
- SOH（Start of Head）：报头开始标志，用于表示报文的标题信息或报头的开始。仅在报文的第一个数据块中才有此标志。
- STX（Start of test）：文本开始标志，标识标题信息的结束和报文文本的开始。每个数据块均有此标志。
- SYN（Synchronous）：字符同步标志，用以实现通信双方的字符同步或用于在无数据传输时保持同步。在每个数据块中均有此标志，而且通常是两个。

以上这十种通信控制字符所对应的 ASCII 码（ASCII 中是用低 7 位表示一个字符的，最高位为校验码）或 EBCDIC 码值如表 6-2 所示。这种通信控制字符中，在数据同步传输中起关键作用的就是 SYN、SOH、STX、ETB、ETX、EOT 这 6 种通信控制字符。

表 6-2　BSC 协议十种通信控制字符对应的代码

控制字符名称	对应的 ASCII 码	对应的 EBCDIC 码
ACK	0000110	00101110
DLE	0010000	00010000
ENQ	0000101	00101101
EOT	0000100	00110111
ETB	0010111	00100110
ETX	0000011	00000011
NAK	0010101	00111101
SOH	0000001	00000001
STX	0000010	00000010
SYN	0010110	00110010

通过前面的介绍可以知道，根据同步传输中报文在数据块中所处的位置不同，BSC 协议的数据块也有所不同，具体有如下 4 种格式：

- 不带报头的单块报文或分块传输中的最后一块报文的格式为：

SYN	SYN	STX	报文	ETX	BCC

- 带报头的单块报文的格式为：

SYN	SYN	SOH	报头	STX	报文	ETX	BCC

- 分块传输中的第一块报文的格式为：

| SYN | SYN | SOH | 报头 | STX | 报文 | ETB | BCC |

- 分块传输中的中间报文的格式为：

| SYN | SYN | STX | 报文 | ETB | BCC |

从以上 4 种数据报文格式可以得出 BSC 报文传输的如下两个基本特点：①BSC 协议中所有发送的数据均跟在至少两个 SYN 字符之后，以使接收端能实现字符同步；②所有数据块在块终止标志符（ETX 或 ETB）之后还有块校验字符 BCC（Block Check Character），是单字节的 CRC（循环校验码）或双字节的 CRC，校验范围从 STX 开始到 ETX 或 ETB 为止。

6.6.2 BSC 协议数据透明传输原理

面向字符的同步传输协议不像异步传输协议那样需要在每个字符前后附加起始和停止位，因此传输效率提高了。但由于采用了一些特殊的传输控制字符，在增强了通信控制能力和校验功能的同时也带来了新的问题，那就是如何区别数据字符代码和特殊控制字符代码。因为在数据块中完全有可能出现与特殊控制字符代码相同的数据字符，这就会在接收端产生误解。比如正文有个与终止字符 ETX 的代码相同（在 ASCII 码中为 0000011）的数据字符，如果不作任何处理，接收端就不会把它当作普通数据处理，而误认为是正文结束，产生差错。

为解决这一问题，面向字符的同步传输协议应具有将数据块中的特殊字符作为普通数据处理的能力，这种能力叫做“透明传输”。BSC 协议的透明传输解决方案是在同步传输协议中设置了转义字符 DLE（Data Link Escape，数据链路封装）。当需要在数据块中传输一个与某个特殊字符代码一样的数据时，就要先在它前面加一个 DLE 代码（ASCII 码中为 0010000），这样接收端在收到一个 DLE 代码时，就知道了它后面一个字符是普通的数据字符，而不是通信控制字符。在接收端如果发现一个与某个特殊字符代码相同的字符是 DLE 代码，则直接删除前面的 DLE 代码，仅接收真实的数据部分。

如果要在数据中传输 EOT 字符，为了避免把它误认为是通信控制字符，就需要先键入一个 DLE 字符，然后再键入 EOT 字符，如图 6-21 上图所示。在接收端会直接删除 EOT 字符代码前面的 DLE 代码，仅接收数据部分的 EOT 代码。

同样，当要传输的数据中有一个与 DLE 相同的数据字符时，也要先在它前面加上另外一个 DLE 字符，相当于要传输一个普通的 DLE 数据字符，就需要键入两个 DLE 字符，如图 6-21 下图所示。在接收端如果发现有连续两个 DLE 字符，则直接删除前面那个 DLE 字符，仅接收后面一个作为数据的 DLE 字符。

从以上可以看出，这种面向字符的同步方法实现起来相当麻烦，因为 BSC 这类同步传输协议中的控制字符本身比较多，很容易就造成了与数据中传输数据代码的冲突。也正是因为这个缺点，后面又产生了下节将要介绍的“面向比特”的同步传输协议。

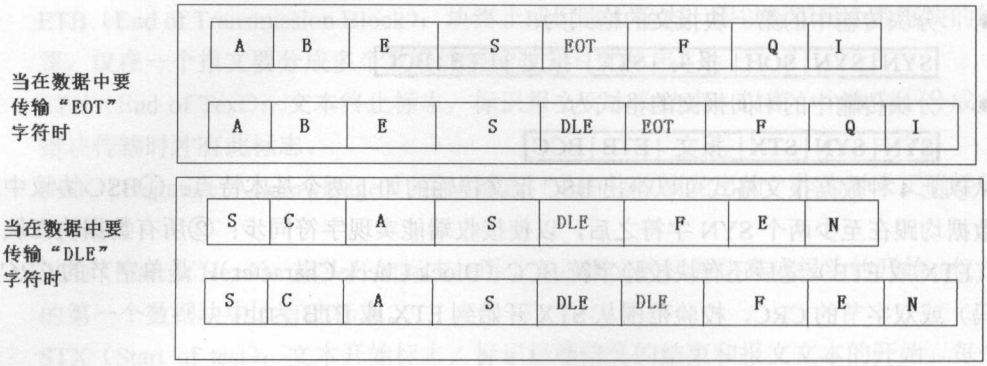


图 6-21　BSC 协议的两个数据透明传输示例

6.7　面向比特的 SDLC 和 HDLC 同步传输协议

“面向比特”的链路层协议主要有：IBM SDLC、ANSI 通过修改 SDLC 而提出的 ADCCP（Advanced Data Communication Control Procedure，高级数据通信控制规程）、ISO 通过修改 SDLC 而提出的 HDLC（High-level Data Link Control，高级数据链路控制）、CCITT 通过修改 HDLC 而提出的 LAP（Link Access Procedure，链路访问规程）等。

“面向比特”的同步传输协议中数据块是作为比特流来处理（而不是作为字符流来处理）的，所以称之为“面向比特”的同步传输。在“面向比特”的同步传输中，每个数据块的头部和尾部都用一个特殊的比特序列（如 01111110）来标记数据块的开始和结束，这就是面向比特的同步传输协议中的基本成帧原理，或者说是基本的帧同步原理。在局域网中所采用的传输方式都是面向比特流的同步传输方式，由它们各自的介质访问控制协议来定义具体的数据格式（即帧格式）以及相应的介质访问控制方法。

6.7.1　HDLC 链路结构和操作方式

SDLC 协议是一种 IBM 数据链路层协议，适用于系统网络体系结构（SNA）；HDLC 是一个在同步网上传输数据、面向比特的数据链路层协议，是 ISO 对 IBM SDLC 协议的改进和标准化协议。但是 SDLC 属于单链路规程，而 HDLC 属于多链路规程。

SDLC 支持识别两类网络站点：主站点（Primary Station）和从站点（Secondary Station）。主站点控制从站点，主站点轮询从站点是否有数据要发送。也就是说，从站点只有在主站点授权前提下才可以向主站点发送信息，如果一个从站点有数据要发送，当它被主站点识别后才开始数据传输。主站点按照预先确定的顺序选择从站点，一旦选定的从站点已经导入数据，那么它即可进行传输。同时主站点可以建立和拆除链路，并在运行过程中控制这些链路。

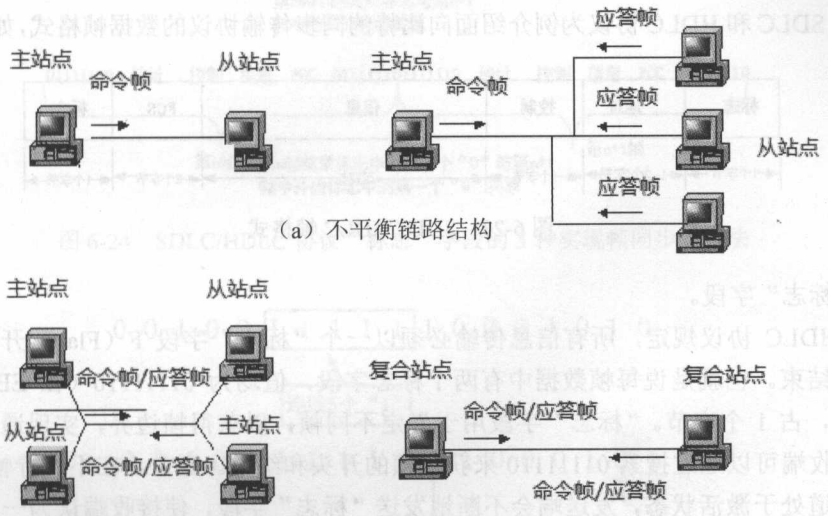
HDLC 是 ISO 对 IBM SDLC 协议的改进版本，定义了以下 3 种站点：

- 主站点：在链路上起控制作用的站点，负责对数据流进行组织，并对链路上的差错实施

恢复，发出的是“命令”帧。主站点需要比从站点具有更多的功能，当终端与主机相连时主机一般总是主站点。

- 从站点：受主站点控制的站点，只能对主站点的“命令”帧进行响应，发出的是“响应”帧。
- 复合站点：同时具备主站点和从站点的双重功能，根据需要既可发出“命令”帧，又可发出“响应”帧。在一个站点连接多条链路的情况下，该站点对于一些链路可能是主站点，而对另外一些链路又可能是从站点。

在 HDLC 中，对主站点、从站点和复合站点定义了如图 6-22 所示的 3 种链路结构：不平衡链路结构、对称不平衡链路结构和平衡链路结构。从中可以看出，不同链路结构，允许的站点类型以及各站点发送命令帧和响应帧的权限也不一样。



（a）不平衡链路结构

（b）对称不平衡链路结构　　（c）平衡链路结构

图 6-22　HDLC 的 3 种链路结构

另外，根据通信双方的链路结构和传输响应类型，HDLC 又定义了 3 种操作方式：正常响应方式、异步响应方式和异步平衡方式。

（1）正常响应方式（NRM）。

正常响应方式（NRM）适用于不平衡链路结构，主要用于点对点和点对多点的链路结构中，特别是点对多点链路，一端为主站点，另一端为从站点，如图 6-22（a）所示。在这种方式中，主站点控制着整个链路的操作，负责链路的初始化、数据流控制和链路复位等，而从站点仅可在收到主站点的明确允许后才能发出响应，所以它们的角色是不对称的，也就是不平衡的。

（2）异步响应方式（ARM）。

异步响应方式（ARM）也适用于不平衡链路结构，但它一般是对称不平衡链路结构，如图 6-22（b）所示。ARM 与 NRM 不同的是：在 ARM 方式中，从站点可以在没有得到主站点允许的情况

下开始数据传输，所以它的传输效率比 NRM 有所提高。

（3）异步平衡方式（ABM）。

异步平衡方式（ABM）适用于平衡链路结构，即链路两端都是复合站点，也就是该站点既可作为主站点，又可作为从站点，如图 6-22（c）所示。在这种链路结构中，两端的复合站点具有同等的能力，不管哪个复合站点均可在任意时间发送命令帧，并且不需要收到对方复合站点发出的命令帧就可以发送响应帧。ITU-T X.25 建议的数据链路层就采用这种方式。

除了以上 3 种基本操作方式外，HDLC 还有 3 种扩充方式，即扩充正常响应方式（SNRM）、扩充异步响应方式（SARM）、扩充异步平衡方式（SABM），它们分别与基本方式相对应。

6.7.2 SDLC/HDLC 帧结构

下面以 SDLC 和 HDLC 协议为例介绍面向比特的同步传输协议的数据帧格式，如图 6-23 所示。

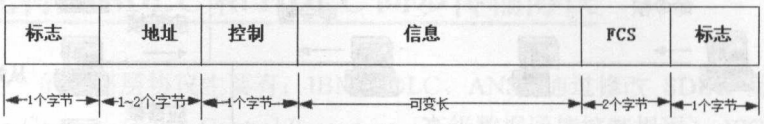


图 6-23　SDLC/HDLC 帧格式

（1）“标志”字段。

SDLC/HDLC 协议规定，所有信息传输必须以一个“标志”字段 F（Flag）开始，且以同一个标志字段结束。也就是说每帧数据中有两个标志字段，值均为 01111110（在 EBCDIC 码中是“=”字符），占 1 个字节。“标志”字段用于界定不同帧，以获得帧边界，实现通信双方的帧同步，因为接收端可以通过搜索 01111110 来获知帧的开头和结束。通常，在不进行帧传送的时刻，为了维持信道处于激活状态，发送端会不断地发送“标志”字段，使接收端认为一个新的帧传送已经开始。

SDLC/HDLC 中使用“标志”字段的方法可以是一个帧包括一个开始标志和一个结束标志（如图 6-24 上图所示），也可以把一个帧的结束标志当成下一个帧的开始标志，也就是共享“标志”字段（如图 6-24 中图所示），还可以把一个帧的结束标志中的最后一个“0”当成下一帧开始标志的第一个“0”（如图 6-24 下图所示）。

在一串数据比特中，有可能产生与“标志”字段的码型相同的比特组合——01111110。为了防止这种情况产生，保证对数据的透明传输，采取了比特填充技术。就是在信号码中出现连续 5 个“1”以后插入一个“0”，如图 6-25 所示。在接收端只需要再去掉 5 个“1”后面插入的“0”即可恢复原来的信号码序列。

（2）“地址”字段。

在“标志”字段之后是一个“地址”字段 A（Address），表明帧来自于主站点还是从站点。主站点或者复合站点发送的命令帧中“地址”字段携带的是对方从站点的地址，而从站点发出的响应帧

的“地址”字段所携带的是本站点的地址。总体来说，在使用不平衡方式（包括 NRM 和 ARM）传输数据时，“地址”字段总是写入从站点的地址；在使用平衡方式（ABM）时，“地址”字段总是写入响应站的地址。

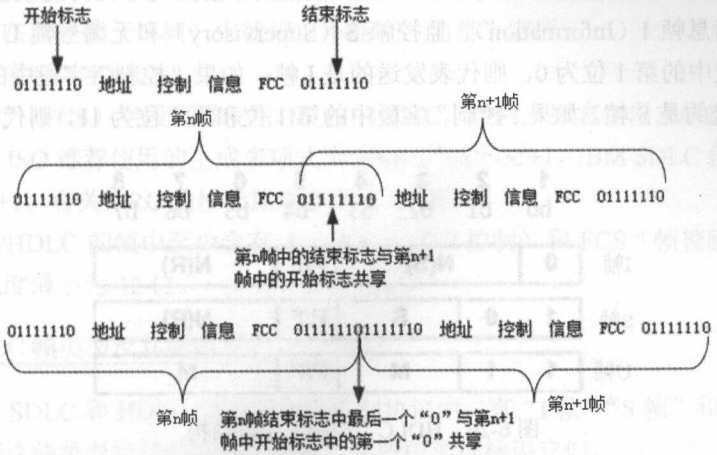


图 6-24 SDLC/HDLC 协议“标志”字段的 3 种实现帧同步的方法

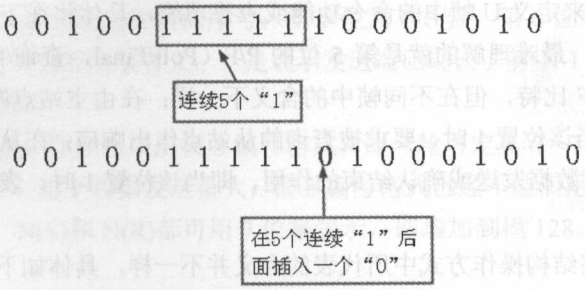


图 6-25 SDLC/HDLC 的数据透明传输示例

“地址”字段占 8 位或 16 位（也就是 1 或 2 个字节），通常是采用 8 位长度。00000000 地址为空地址，不能分配给任何站点，用于测试数据链路的状态；11111111 地址为广播地址，使用这个地址就可以把一个数据帧发送到同一网段中的其他所有站点，类似于以太网中的广播发送方式。因此在采用 8 位地址格式时，总的有效地址数是 254 个，这对一般的多点链路是足够的。但考虑在某些情况下，例如使用分组无线网，用户可能很多，可使用扩充地址字段，以字节为单位扩充。在扩充时，每个地址字段的第 1 位用作扩充指示，即当第 1 位为“0”时，表示后续一个字节为扩充地址字段；当第 1 位为“1”时，表示后续一个字节不是扩充地址字段，地址字段到此为止。

（3）“控制”字段。

“控制”字段 C（Control）用来实现 HDLC 协议的各种控制信息，并标识是否是数据（因为

它可以标识不同的帧类型），占 1 个字节长度。发送方主站点或复合站点利用“控制”字段来通知被寻址的从站点或复合站点执行约定的操作；而从站点用该字段对来自主站点或复合站点的命令帧进行响应，报告已完成的操作或状态的变化。

“控制”字段的结构如图 6-26 所示，共 8 位，根据其最前面两个比特的取值可将 HDLC 帧划分为三大类，即信息帧 I（Information）、监控帧 S（Supervisory）和无编号帧 U（Unnumbered）。如果“控制”字段中的第 1 位为 0，则代表发送的是 I 帧；如果“控制”字段中的第 1 位和第 2 位为 10，则代表发送的是 S 帧，如果“控制”字段中的第 1 位和第 2 位为 11，则代表发送的是 U 帧。

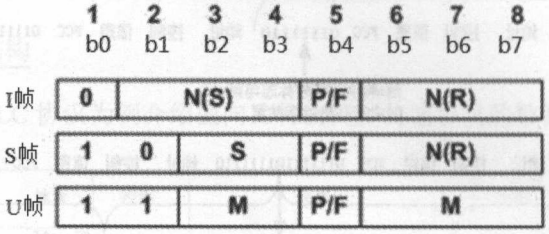


图 6-26　HDLC“控制”字段结构

N(S)代表所发送的帧的编号，N(R)代表希望下次从对端接收的帧编号，表示在此编号前的所有帧本端已正确接收。这类似于第 10 章中介绍 TCP 协议数据结构时将要介绍的 ACK 编号。S 用来定义 S 帧的编码，M 用来定义 U 帧中的命令功能或应答功能，具体将在下节介绍。

在“控制”字段中，最难理解的就是第 5 位的 P/F（Poll/Final，查询/结束）位。在 HDLC 的各类帧中均带有这个 P/F 比特，但在不同帧中的含义不一样：在由主站点发出的命令帧中取“P”位，起查询的作用，即当该位置 1 时，要求被查询的从站点作出响应；在从站点响应主站点的响应帧中取“F”位，起结束数据发送或确认结束的作用，即当该位置 1 时，表示从站点数据发送完毕或者响应完毕。

P/F 位在不同的链路结构操作方式中所代表的含义并不一样，具体如下：

- 在 NRM 方式中，从站点不能主动向主站点发送信息，从站点只有在收到主站点发出的 P 比特为 1（表示对各从站点依次进行查询，看看这些从站点是否有数据发送）的命令帧以后才能发送对应的响应帧：如果从站点有数据发送，则在最后一个 I 帧中将 F 比特置 1（表示数据发送结束），中间的 I 帧 F 比特置 0（表示数据还没发完）；如果从站点无数据发送，则要向主站点发送将 F 比特置 1 的响应帧。
- 在 ARM 或 ABM 方式中，任何一个站点都可以在主动发送的 S 帧和 I 帧中将 P 比特置 1（同样是表示对各从站点或者对方复合站点依次进行查询，看看它们是否有数据发送）。对方站点在收到 P=1 的 S 帧或 I 帧后，同样会按照上面介绍的方法进行处理：如果对方站点有数据发送，则在最后一个 S 帧或 I 帧中将 F 比特置 1（表示数据发送结束），中间的 S 帧或 I 帧 F 比特置 0（表示数据还没发完）；如果对方站点无数据发送，则要向主站点发送将 F 比特置 1 的响应帧。

（4）“信息”字段。

“信息”字段 I（Information）包含了用户的数据信息和来自上层的各种控制信息。在 I 帧和某些 U 帧中具有该字段，而且可以是任意长度的比特序列。在实际应用中，其长度由收发站缓冲器的大小和线路的带宽情况决定，但必须是 8 比特的整数倍。该字段可以是 0 长度，也就是可以无“信息”字段，如在监控帧（S 帧）中就规定不能有“信息”字段。

（5）“帧校验序列”字段。

“帧校验序列”（FCS）字段可以使用 16 位的 CRC，对两个标志字段之间的整个帧的内容进行校验。CCITT 和 ISO 推荐使用的生成多项式为 $g(x)=x^{16}+x^{12}+x^5+1$，IBM SDLC 使用的生成多项式为 $g(x)=x^{16}+x^{15}+x^2+1$。有关 CRC 的校验原理参见 6.3.2 节。

由于在 SDLC/HDLC 的帧中至少含有 A（地址）、C（控制）和 FCS（帧校验序列）这 3 个字段，所以整个帧长度最小为 32 位。

6.7.3 SDLC/HDLC 帧类型及其标识方法

前面说了，在 SDLC 和 HDLC 等数据链路控制协议中，有“I 帧”“S 帧”和“U 帧”3 种帧类型。下面具体介绍这些类型的帧以及在“控制”字段中如何标识它们。

1. 信息帧（I 帧）

I 帧用于用户数据传输，包含“信息”字段。在 I 帧“控制”字段中第 1 位固定为 0；第 2 位至第 4 位为 N(S)，用于标识要发送的帧的序号；第 5 位为 P/F 位；第 6 位至第 8 位为 N(R)，用于标识期待要接收的帧序号（它有两层含义：一是表示发送端已确认了前 N-1 个帧，另一个是发送端期待发送第 N 帧的确认帧）。

N(S)和 N(R)各占 3 位，即序号采用模 8 编号格式，即一轮可定义 0～7 八个编号的数据帧。在有些场合，如卫星通信中，由于数据发送量大，模 8 编号格式已经不能满足要求了，这时可以把控制字段扩展为两个字节，N(S)和 N(R)都可用 7 位来表示，即增加到模 128。

2. 监控帧（S 帧）

S 帧用于监视和控制数据链路，完成 I 帧的接收确认、重发请求、暂停发送请求等功能。S 帧没有“信息”字段（一共只有 48 位），可由主站点或从站点发送，具体要看对应的链路结构操作方式。在 S 帧的“控制”字段中第 1 位和第 2 位分别固定为 1、0，第 3 位和第 4 位为 S，代表监控功能，第 5 位 P/F 和第 6 位至第 8 位与 I 帧一样。

因为用于表示监控功能的 S 共有 2 位，所以可以代表 4 种监控类型的帧，具体如下：

- 00：接收就绪（RR），主站点可以使用 RR 型 S 帧来查询从站点，即希望从站点传输“控制”字段中第 6 位至第 8 位 N(R)所指示的编号为 N(R)的 I 帧，如果从站点存在这样的帧，便进行传输。从站点也可用 RR 型 S 帧来作为对主站点的响应帧，表示从站点希望从主站点那里接收的下一个 I 帧的编号是 N(R)。
- 01：拒绝（REJ），用于接收端要求发送端对从编号为 N(R)开始的帧及其以后所有的帧进行重发，这也暗示 N(R)以前的 I 帧已被正确接收。

- 10：接收未就绪（RNR），用于接收端通知发送端编号小于N(R)的I帧已被收到，但目前正处于忙状态，尚未准备好接收编号为N(R)的I帧，这可用来对链路流量进行控制。
- 11：选择拒绝（SREJ），用于接收端要求发送端发送编号为N(R)的单个I帧，并暗示其他编号的I帧已全部确认。

可以看出，接收就绪RR型S帧和接收未就绪RNR型S帧有两个主要功能：首先，这两种类型的S帧用来表示从站点已准备好或未准备好接收信息；其次，确认编号小于N(R)的所有I帧均已正确接收。拒绝REJ和选择拒绝SREJ型S帧用于向对方站指出发生了差错。REJ帧用于“回退N帧”策略，用以请求重发N(R)以前的帧已被确认，当收到一个N(S)等于REJ型S帧的N(R)的I帧后，REJ状态即可清除。SREJ帧用于“选择重发”策略，当收到一个N(S)等SREJ帧的N(R)的I帧时，SREJ状态即应消除。

3. 无编号帧（U帧）

U帧用于数据链路的控制，不带编号，可以在任何需要的时刻发出，而不影响带编号的信息帧的交换顺序。在U帧的“控制”字段中第1位和第2位都固定为1，第3位和第4位以及第6位至第8位均为M，代表无编号功能，第5位P/F与I帧一样。

U帧可以分为命令帧（C）和响应帧（R），通过5M比特位来表示不同功能，具体如表6-3所示。

表6-3　无编号帧的类型及编码

帧名称	功能	帧类型		M比特位				
		命令帧	响应帧	b_2	b_3	b_5	b_6	b_7
SNRM	置正常响应模式	C		0	0	0	0	1
SARM/DM	置异步响应模式/断开方式	C	R	1	1	0	0	0
SABM	置异步平衡模式	C		1	1	1	0	0
SNRME	置扩充正常响应模式	C		1	1	0	1	1
SARME	置扩充异步响应模式	C		1	1	0	1	0
SABME	置扩充异步平衡模式	C		1	1	1	1	0
DISC/RD	断链/请求断链	C	R	0	0	0	1	0
SIM/RIM	置初始化方式/请求初始化方式	C		1	0	0	0	0
UP	无编号查询	C		0	0	1	0	0
UI	无编号信息	C		0	0	0	0	0
XID	交换识别	C	R	1	1	1	0	1
RESET	复位	C		1	1	0	0	1
FRMR	帧拒绝		R	1	0	0	0	1
UA	无编号确认		R	0	0	1	1	0

6.8 面向字符的 PPP 同步传输协议

PPP（Point-to-Point Protocol，点对点协议）是目前应用非常广泛的一种广域网数据链路层协议。例如我们在使用 Modem、ADSL Modem 进行拨号连接时都需要用到它（ADSL Modem 中的 PPPoE 协议的底层也是 PPP 协议），路由器设备间的 Serial 口之间的连接，还有在像 SDH（同步数字系列）网络中都要封装这个协议。

6.8.1 PPP 简介

在点对点链路上，最早使用的数据链路层协议不是 PPP，而是 SLIP（Serial Line Internet Protocol，串行线路网际协议），如 Windows 98 系统中 Modem 拨号就是使用 SLIP 而不是 PPP。但 SLIP 具有以下这些根本无法适应当时网络技术发展和应用需求的天生不足：

（1）连接速率低。

SLIP 使用的线路速率一般介于 1200b/s 和 19.2Kb/s 之间，远没有 PPP 的连接速率高（单 PPP 信道可达 128Kb/s，通过一些复用技术可高达好几 Gb/s）。

（2）不能自动分配 IP 地址。

进行 SLIP 连接的通信双方必须先具备静态 IP 地址，不能在连接过程中动态分配 IP 地址。所以当时的 SLIP 通常是应用于专线连接中，在拨号连接中也仅应用于固定 IP 地址方式的连接。

（3）无“协议”（Protocol）类型字段。

在 SLIP 帧中没有协议类型字段，只支持默认的 IP 协议。SLIP 帧很简单，只是在 IP 包的最前面和最后面各加一个 END 字符（0xc0）作为帧边界，标识一个帧的起始和结束。如果在包中有 END 字符，则包中的 END 字符用 0xdb（ESC 字符）和 0xdc 两个字符来替代；如果在包中有 ESC 字符，则 ESC 字符用 0xdb（ESC 字符）和 0xdd 两个字符来替代，以实现数据透明传输，如图 6-27 所示。

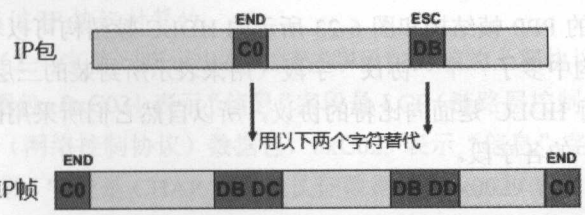


图 6-27 SLIP 帧格式及透明传输示意图

（4）无“帧校验序列”（FCS）字段。

从图 6-27 可以看出，SLIP 帧中没有 FCS 字段，因此无差错检测能力，必须由上层实体或具有纠错能力的 Modem 来解决差错问题。

PPP 协议是在 SLIP 的基础上发展起来的，对 SLIP 的以上问题均加以了解决，但仍是“面向字符”的链路层协议。具体来讲，PPP 协议具有以下几个方面的特性：

（1）PPP 在连接速率上可以远高于 SLIP，可以最高达到 128Kb/s，如像 v.90 以上标准的 Modem 都可达到 64Kb/s。

（2）提供了“协议”（Protocol）类型字段和“帧校验序列”（FCS）字段（如图 6-28 所示），使得 PPP 除了支持 IP 协议包封装外，还可以封装其他三层协议包，如当时 DECnet 和 Novell 的 Internet 网包交换（IPX）。另外，有了 FCS 字段，就可以提供各种差错控制功能。

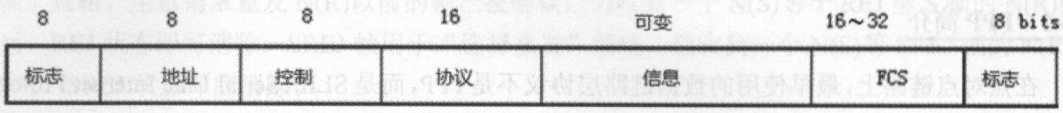


图 6-28　PPP 帧格式

（3）提供了一整套方案来解决链路建立、维护、拆除、上层协议协商、认证等问题。这些功能是通过以下几个子协议来完成的：

- 链路控制协议（Link Control Protocol，LCP）：用于建立、配置、测试和管理数据链路连接。
- 网络控制协议（Network Control Protocol，NCP）：协商该链路上所传输的数据包格式与类型，建立、配置不同的网络层协议。
- 口令认证协议（Password Authentication Protocol，PAP）和质询握手认证协议（Challenge-Handshake Authentication Protocol，CHAP）：为 PPP 连接提供用户认证功能，可以确保 PPP 连接的安全性。

PPP 协议的简单和功能完整特点使它得到了广泛的应用，相信在未来的网络技术发展中还可以发挥更大的作用。

6.8.2　PPP 帧结构和透明传输原理

比较图 6-28 所示的 PPP 帧结构和图 6-23 所示的 HDLC 帧结构可以看出它们是很相似的，其主要区别是 PPP 帧结构中多了一个“协议”字段（用来表示所封装的三层协议包类型），而且 PPP 是面向字符的协议，而 HDLC 是面向比特的协议，所以自然它们所采用的填充方式也不一样。下面先介绍 PPP 帧结构中的各字段。

1. PPP 帧结构

PPP 帧结构参见图 6-28，共分 7 个字段，其中“标志”字段在帧的最前面和最后面均有一个，其他字段各一个。下面是这些字段的具体含义说明。

（1）标志（Flag）。

用来标识帧的起始或结束，占 8 位（1 个字节），值固定为 01111110（0x7E），与 HDLC 帧中的“标志”字段的值是一样的。

（2）地址（Address）。

本来是用来标识对方节点地址的，但因 PPP 是点对点通信协议，是明确知道对方节点的，在实际通信中是无需知道对方的数据链路层地址（也就是 MAC 地址）的，从实际通信角度考虑，此“地址”字段实际上是没什么意义的，在 PPP 帧中此“地址”字段为固定的 11111111（0xFF）标准广播地址，占 8 位（1 个字节）。这与 HDLC 中的“地址”字段是不一样的。

（2）控制（Control）。

因为 PPP 本身是一种可靠的点对点数据链路层通信协议，无需像 HDLC 协议那样需要额外提供可靠的链路连接服务，所以 PPP 只有一种帧类型，那就是 UI（无编号信息）帧。因为它是无编号的帧，所以也就无需接收端对收到的帧进行确认，不能利用帧编号进行差错控制。也正因为如此，PPP 帧中的“控制”字段其实也没有意义，值固定为 00000011（0x03）。

（2）协议（Protocol）。

PPP 帧与 HDLC 帧的最大区别就是 PPP 帧中有“协议”字段，而 HDLC 帧中没有。之所以 PPP 帧中有“协议”字段，是因为它除了可以封装 IP 协议外，还可封装其他多种网络层协议，如 IPX、AppleTalk 等。但这里不仅指网络层协议，还指与 PPP 协议同样在数据链路层，但是其内部的层次高于 PPP 的其他协议，如前面提到的 LCP、NCP、PAP 和 CHAP 子协议。



在体系结构中的每一层包括了许多通信协议，但一些层次中的这些协议在本层内部又有层次高低之分，而且高层次的协议报文必须经由低层次的协议封装。如这里的 LCP、NCP、PAP、CHAP 与 PPP 协议都同在数据链路层中，但它们在数据链路层内部的层次是高于 PPP 协议的，所以它们的报文在发送端必须经过 PPP 协议封装。

在本书后面将要介绍的网络层、传输层同样存在这样的情况。如 ICMP、IGMP 等协议与 IP 协议同位于网络层，但在网络层内部这些协议的层次又高于 IP 协议，所以 ICMP、IGMP 等协议报文需要经过 IP 协议封装。但是同处于网络层的 ARP 协议，因为它在网络层中的层次比 IP 协议低，所以 ARP 报文是不需要经过 IP 协议封装的。

“协议”字段占 16 位（2 个字节），指示出在“信息”字段中封装的上层协议数据类型，如 0x0021 表示“信息”字段是 IP 数据包，0xC021 表示“信息”字段是 LCP（链路层控制协议）数据包，0x8021 表示“信息”字段是 NCP（网络控制协议）数据包，0xC023 表示“信息”字段是 PAP 安全性认证数据包，0xC223 表示“信息”字段是 CHAP 安全性认证数据包，0x0029 表示“信息”字段为 Apple Talk 协议数据包，……。

（5）信息（Information）。

来自上层协议（包括网络层协议，也包括在数据链路层中比 PPP 协议层次更高些的协议）的有效数据，可以是任意长度，默认为 1500 个字节，如果不够该长度，还可以通过填充方法达到这个长度。

（6）帧校验序列（FCS）。

使用 16 比特 CRC 计算得出的“信息”字段校验和，用于在接收端对所接收的数据进行正确性验证。

2. 透明传输

从前面介绍的 PPP 帧结构可以看出，在帧的首、尾均有一个用于标识帧边界的“标志”字段，其值均固定为 01111110（0x7E），这就同样要面对一个问题，那就是当在“信息”字段中出现和“标志”字段一样的比特 0x7E 时，接收端可能误把这些比特当成帧边界。

为了解决这个问题，也就是实现透明的数据传输就必须采取一些措施。但因为 PPP 是“面向字符”协议，所以它不能采用 HDLC 所使用的零比特插入法，而是使用一种特殊的字符（也就是前面所说的“转义字符”）-0x7D 进行填充。具体的做法是将“信息”字段中出现的每一个 0x7E 字节转变成 2 字节序列(0x7D,0x5E)，即 01111101 01011110；如果“信息”字段中出现一个 0x7D 字节，则要将其转变成 2 字节序列(0x7D,0x5D)，即 01111101 01011101；如果“信息”字段中出现 ASCII 码的控制字符（如值为 0x27 的 ESC 字符），则在该字符前面要加入一个 0x7D 字节，如图 6-29 所示。这样做的目的是防止这些表面上的 ASCII 码控制字符被错误地解释为控制字符。

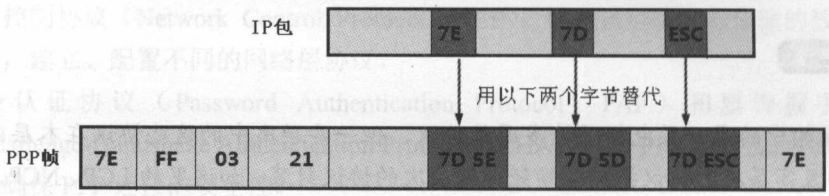


图 6-29　PPP 帧格式及透明传输示意图

6.8.3　PPP 链路建立、使用和拆除流程

在 PPP 点对点通信中，因为不像局域网中的链路那样是始终连接的，所以在建立 PPP 通信前，通信双方必须协商建立链路连接，在链路建立后才可进行数据传输，数据传输完成后又可拆除原来建立的链路。

整个 PPP 链路的建立过程需要用到前面介绍的三组子协议：LCP、NCP 和 PAP 或 CHAP 认证协议。整体过程分为 5 个阶段：Dead（死亡）阶段、Establish（链路建立）阶段、Authenticate（身份认证）阶段、Network（网络控制协商）阶段和 Terminate（结束）阶段。不同阶段进行不同协议的协商，只有前面的协议协商出结果后才能转入下一个阶段协议的协商，如图 6-30 所示。具体流程如下：

（1）当有用户向 ISP 或者对端节点发起 PPP 连接请求时，首先打开封装了 PPP 协议的物理接口，然后通过封装了 LCP 子协议的 PPP 帧与对端 PPP 接口进行协商，协商内容包括工作方式是 SP（单 PPP 通信）还是 MP（多 PPP 通信）、认证方式（可以没有认证的）和链路最大传输单元（MTU）等。

（2）LCP 协商完成后就进入 Establish 阶段，进行数据链路建立。这时主要是启用 PPP 协议对接口进行封装，与对端 PPP 接口建立 PPP 连接，如果在 PPP 接口上没有配置身份认证，并且 PPP 连接建立成功，则在链路建立成功后保持 LCP 为激活状态。如果 PPP 连接建立不成功，则关闭 LCP 子协议，关闭 PPP 接口。

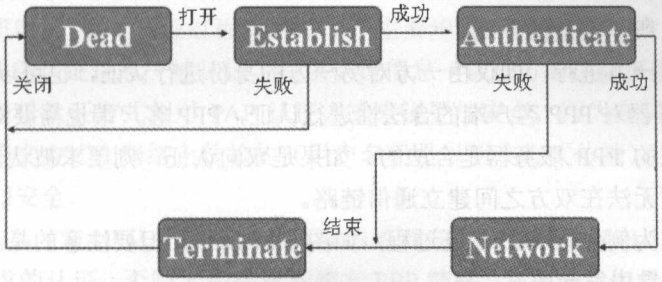


图 6-30　PPP 链路建立、使用和拆除流程

（3）如果 PPP 链路一端或两端接口上配置了 PPP 认证，则需要进入 Authenticate 阶段，对请求连接的用户进行身份认证。但这时必须确保链路两端的接口配置了相同的 PPP 认证方式，并且配置正确，否则认证立即失败，关闭 LCP 子协议，关闭 PPP 接口。

（4）如果 PPP 链路两端配置了 PPP 认证且认证成功则进入 Network 阶段，使用封装了 NCP 子协议的 PPP 帧与对应的网络层协议进行协商，主要是网络层地址协商。协商成功就最终建立 PPP 连接，否则先关闭 NCP 子协议，然后再关闭 LCP 子协议，最终关闭 PPP 接口。



PPP 接口网络地址的配置与以太网接口的网络地址配置有较大区别。首先是以太网链路两端的接口 IP 地址必须在同一个 IP 网段，而 PPP 链路两端的接口 IP 可以不在同一个 IP 网段。另外，在 PPP 链路上可以只为一端的接口配置了 IP 地址，另一端可以通过由对端固定指定或者通过地址池动态分配一个 IP 地址，这也是以太网接口所不具备的。

（5）PPP 链路建立好后就可以直接用于网络通信进行数据传输了，直到用户拆除了原来所建立的 PPP 链路，进入到 Terminate 阶段。

PPP 链路的拆除过程与建立过程是相反的，先关闭 NCP 协议，释放原来为用户分配的临时网络层地址，再关闭 LCP 协议，关闭原来建立的 PPP 链路，最后返回到 Dead 阶段。

6.8.4　PPP 的 PAP/CHAP 身份认证

在 PPP 的点对点通信中，可以采用 PAP（口令认证协议）或 CHAP（质询握手认证协议）身份认证方式对连接用户进行身份认证，以防非法用户的 PPP 连接。如果双方达成一致，也可以不采用任何身份认证方式（如一般情况下的路由器间 Serial 口之间的 PPP 连接），但一旦确定要启

用身份认证，则必须确保 PPP 链路两端的接口上启用了相同的 PPP 认证方式，并且相关功能配置正确。

1. PAP 身份认证

PAP 认证过程非常简单，是一个二次握手机制，整个认证过程仅需两个步骤：被认证方（PPP 客户端）发送认证请求→认证方（PPP 服务器）给出认证结果。但是它是以明文方式在线路上传输认证用户名和密码，所以安全性不高。

PAP 认证可以在一方进行，即仅由一方对另一方的身份进行认证，也可以进行双向身份认证，也就是既要 PPP 服务器对 PPP 客户端的合法性进行认证，PPP 客户端也需要对 PPP 服务器进行认证，以确保用于认证的 PPP 服务器是合法的。如果是双向认证，则要求被认证的双方都要通过对方的认证程序，否则无法在双方之间建立通信链路。

下面以单向认证为例介绍 PAP 认证过程，如图 6-31 所示。但要注意的是，PAP 认证是由被认证方（PPP 客户端）首先发起的。

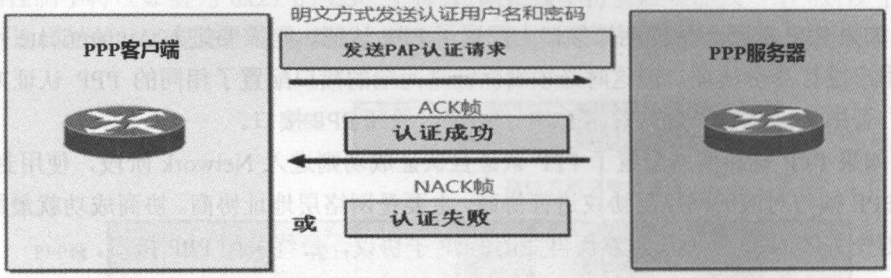


图 6-31　PAP 身份认证的两次握手

（1）发起 PPP 连接的客户端（被认证方）首先以明文方式向 PPP 服务器端（如果是向 ISP 拨号，则 PAP 服务器是在 ISP 端，如果是路由器的串口对连，则 PPP 服务器必须要在对端设备上已配置）发送一个认证请求帧，其中就包括用于身份认证的用户名和密码。

（2）PPP 服务器端（认证方）在收到客户端发来的认证请求帧后，先查看 PPP 服务器本地配置的用户账户数据库，看是否有客户端提供的用户名（这个用户账户数据库必须先在 PPP 服务器端配置好）。如果有，且对应的用户账户密码也一致，则表明客户端具有合法的用户账户信息，向 PPP 客户端返回一个认证确认（ACK）帧，表示认证成功，则该用户可以与 PPP 服务器端建立 PPP 连接。

如果在本地数据库中找不到与客户端发来的用户名一致的用户账户，或者虽然有相同名称的用户账户，但密码不一致，则会认证失败，返回一个认证拒绝（NACK）帧，客户端也不能与 PAP 服务器端建立 PPP 连接。

【注意】如果第一次认证失败，并不会马上直接将链路关闭，而是会在 PAP 客户端提示可以尝试以新的用户账户信息进行再次认证，只有当认证不通过次数达到一定值（默认为 4）时才会关闭链路，以防止因误传、网络干扰等造成不必要的 LCP 重新协商过程。

以上介绍的是 PAP 单向认证过程，仅两步（一问一答形式），很简单。PAP 双向认证过程也与单向认证过程类似，只不过此时 PPP 链路的两端同时具有客户端和服务器双重角色，任何一端都向对方发送认证请求，同时对对方发来的认证请求进行认证。

2. CHAP 身份认证

CHAP 认证过程相对前面介绍的 PAP 认证来说更为复杂，它采用的是三次握手机制（而不是 PAP 中的两次握手机制），整个认证过程要经过 3 个主要步骤：认证方（PPP 服务器）要求被认证方（PPP 客户端）提供认证信息→被认证方提供认证信息→认证方给出认证结果。

另外，CHAP 身份认证方式相对 PAP 认证方式来说更加安全，因为在认证过程中，用于认证的密码不是直接以明文方式在网络上传输的（用户名仍是以明文方式传输），而是封装在 MD5 加密摘要信息中，更加安全。

与 PAP 认证一样，CHAP 认证也可以是单向的或双向的。如果是双向认证，则要求通信双方均要通过对对方请求的认证，否则无法在双方建立 PPP 链路。在此，我们仍以单向认证为例介绍 CHAP 认证流程。具体如图 6-32 所示，但是要注意，CHAP 身份认证首先是由 PPP 服务器端发起的。



图 6-32 CHAP 身份认证的三次握手

（1）当 PPP 客户端要与 PPP 服务器建立连接并且配置采用 CPAP 身份认证方式时，PPP 服务器（认证方）会首先向 PPP 客户（被认证方）端发送一个随机序列号的 MD5 摘要消息进行“质询”（Challenge，也称“挑战”），询问客户端用于身份认证的账户信息。同时这个发送的 MD5 摘要消息会保存在缓存中。

（2）PPP 客户端在收到服务器端发来的质询消息后，把自己要用于身份认证的用户名以明文方式发送，而该账户对应的密码则与收到的来自 PPP 服务器的 MD5 摘要消息进行哈希计算，生成一个加密报文发送给 PPP 服务器。

（3）服务器端在收到来自客户端的响应后，首先直接利用来自 PPP 客户端的明文用户账户名在本地数据库中进行查找，如果找不到该账户，则直接认证失败；如果在本地数据库中有该用户账户就可以得到本地配置的该账户的密码，然后再利用所查到的该账户密码与原来发送给客户端并且

在缓存中保存的 MD5 摘要消息进行哈希计算，用最终的结果与来自 PPP 客户端的 MD5 哈希报文进行比较。

如果两个 MD5 哈希报文完全一致（如果本地配置的该账户密码与在 PPP 客户端配置的一样，则肯定一致，因为此时进行哈希计算的 MD5 报文和密码都是一样的），则认为 PPP 客户端具有合法的用户账户信息，认证通过，向 PPP 客户端发送 ACK 确认帧，成功进行 PPP 连接；否则表示认证失败，向 PPP 客户端发送 NACK 否认帧，不能建立 PPP 连接。

与 PAP 一样，第一次认证失败后，也不会马上关闭链路，而是再次向客户端提示输入新的用户名和密码进行再次认证，直到规定的最高尝试次数。

6.9 数据链路层设备及二层交换原理

介绍完数据链路层的主要功能和服务后，下面再来简单介绍一下工作在数据链路层上的主要网络设备（主要有网卡、网桥和交换机）和用在交换网络中的二层交换原理。

6.9.1 计算机网卡

说到计算机网卡（也叫网络适配器），大家可能认为这没什么好说的，大家都见过，也天天在用。网卡安装在计算机上，用来连接计算机网络，是计算机网络中最基础的网络设备。目前在计算机局域网中，网卡类型总的来说主要可分为有线以太网卡和 WLAN 无线网卡两大类。

1. 有线以太网卡

在企业局域网中目前主要是指以太网卡，有几个方面可以作为划分的标准：一是网卡的主机接口，也就是网卡与计算机连接的接口；二是网卡主机接口总线的位数；三是网卡的网络接口类型，也对应了网卡所支持的传输介质类型；四是网卡所支持的以太网标准。

在主机接口方面，有线网卡主要有以下几种类型：一是最常见的 PCI 总线接口；二是微型 PCI（PCMCIA）；三是在服务器网卡上用得比较多的 PCI-X 和 PCI-E 接口。这几种主机接口类型网卡如图 6-33 所示。

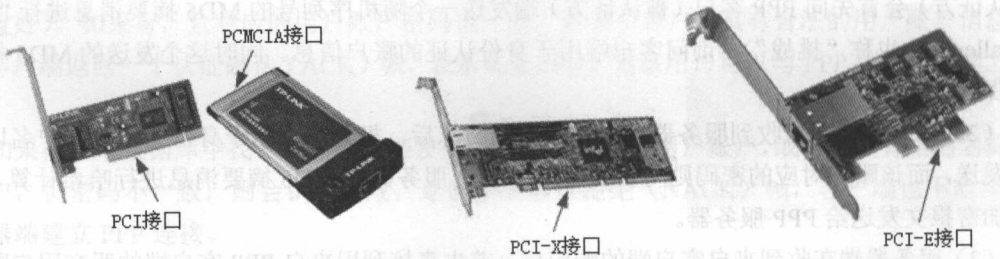


图 6-33 PCI、PCMCIA、PCI-X 和 PCI-E 接口网卡

在有线网卡的网络接口方面，主要对应的是不同的传输介质。因为使用同轴电缆的以太网目前

在企业局域网中比较少见了，所以这里不再介绍粗/细同轴电缆接口的以太网卡了，仅介绍双绞线接口以太网卡和光纤接口以太网卡。

双绞线接口以太网卡使用最普遍，对应双绞线连接器的接口为 RJ-45 类型。RJ-45 连接器（俗称“水晶头”）和网卡上对应的 RJ-45 网络接口如图 6-34 所示。

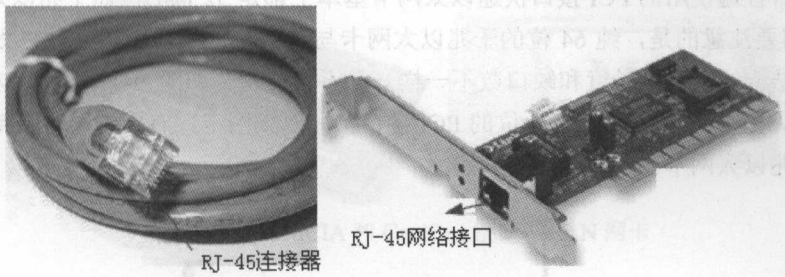


图 6-34　RJ-45 连接器和网络接口

使用光纤作为传输介质的网卡所对应的光纤连接器接口有好几种，最常见的有 ST、SC、FC、LC 这么 4 种，这在第 5 章中已有详细介绍，这里不再赘述。ST、SC、FC 和 LC 这 4 种主要的光纤连接器接口如图 6-35 所示。如图 6-36 所示是目前应用最广的两种光纤网络接口 SC 和 LC 网卡示例。

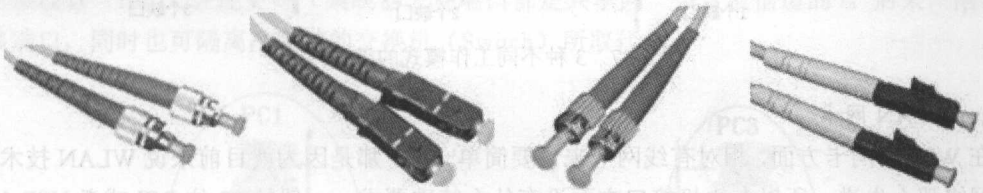


图 6-35　ST、SC、FC 和 LC 光纤连接器

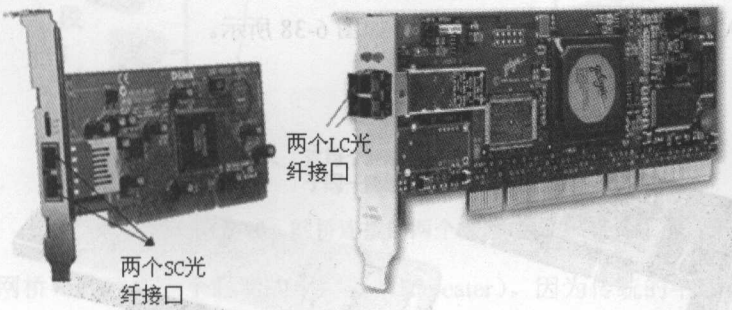


图 6-36　SC 和 LC 网络接口网卡示例

有线网卡除了可以按主机接口和网络接口划分外，还可以根据网卡所支持的网络标准来划分。目前，在有线以太网工作站中我们通常是采用支持 10/100Mb/s 自适应的双绞线快速以太网卡，性

能要求高一些的可能会用到支持 10/100/1000Mb/s 自适应的双绞线千兆以太网卡。而服务器上的网卡目前基本上都是 10/100/1000Mb/s 自适应的双绞线千兆以太网卡或者纯 1000Mb/s 的光纤千兆以太网卡。

另外，支持的这些不同网络标准的 PCI 接口以太网卡所对应的主机接口工作模式也会有所不同。工作站上所普遍使用的 PCI 接口快速以太网卡基本上都是 32 位的，而千兆以太网卡基本上都是 64 位的。需要注意的是，纯 64 位的千兆以太网卡与向后兼容 32 位的 64 位以太网卡的 PCI 总线接口金手指结构（金手指长度和缺口数不一样）是不一样的。如图 6-37 所示，左图是 32 位的 PCI 接口快速以太网卡，中图为纯 64 位的 PCI 接口千兆以太网卡，右图为同时支持 32 位和 64 位的 PCI 接口千兆以太网卡。

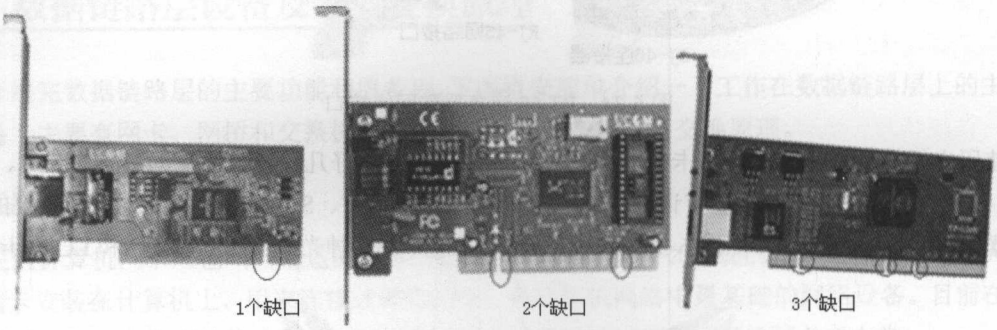


图 6-37　3 种不同工作模式的以太网卡

2. WLAN 网卡

在 WLAN 网卡方面，相对有线网卡来说要简单一些，那是因为就目前来说 WLAN 技术还不如有线网络那么先进，所以在主机接口方面没有什么特殊要求，一般的 32 位 PCI 或者 USB 1.1、2.0 或 3.0 版本接口即可满足。

主要考虑网卡的主机接口和所使用的 WLAN 技术标准。在台式机工作站中通常选用 PCI 或 USB 接口的 WLAN 网卡（主要是 PCI 接口），如图 6-38 所示。

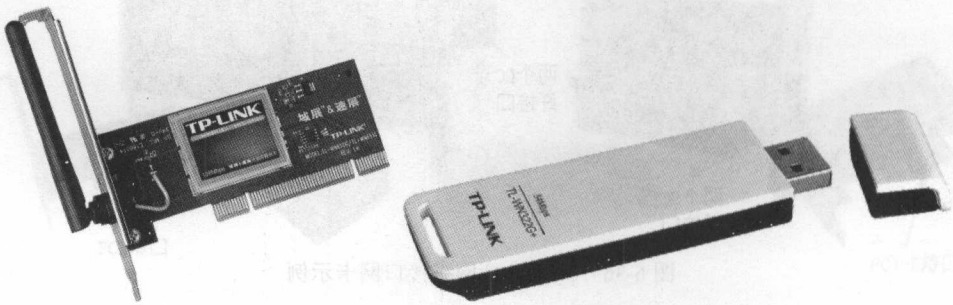


图 6-38　PCI 和 USB WLAN 网卡

笔记本用户则可以选择 PCMCIA 和 USB 两种接口类型的无线局域网网卡。而 PCMCIA 接口

又分 16 位的 PCMCIA 和 32 位的 CardBus 两种接口类型，如图 6-39 所示。在无线局域网标准上，目前至少建议应选择具有 54Mb/s 速率的 IEEE 802.11g 标准无线网卡产品，普遍采用的是支持 600Mb/s 速率的 IEEE 802.11n 标准无线网卡。

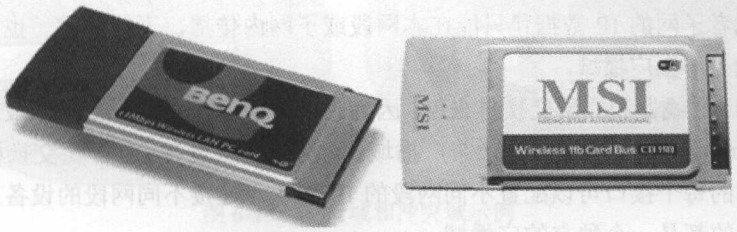


图 6-39 PCMCIA 和 CardBus 接口 WLAN 网卡

6.9.2 网桥及广播域、冲突域

网桥（Bridge）是早期的两端口（目前新的网桥端口数有所增加）二层交换设备，主要用来连接不同网段的计算机网络设备，如图 6-40 所示（因为端口少，一般不是直接用来连接两台计算机的），同时它又可以隔离冲突域，因为它的两个端口在设备背板中分别有一条独立的交换信道，比当时的集线器（Hub）性能更好（集线器上各端口都是共享同一条背板信道的）。后来，网桥被具有更多端口，同时也可隔离冲突域的交换机（Switch）所取代。

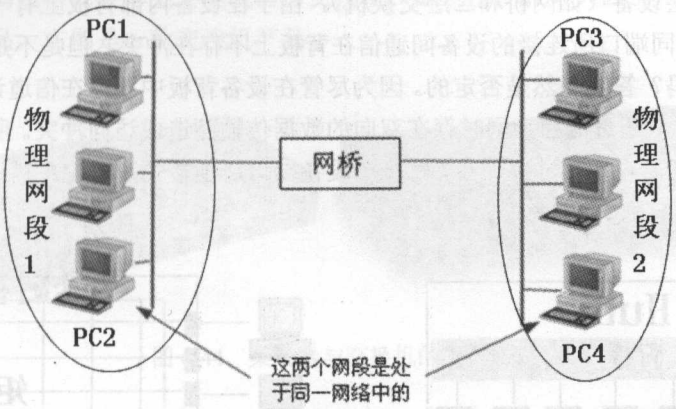


图 6-40 网桥连接的两个物理网段

也有人把“网桥”比喻成一个聪明的中继器（Repeater）。因为传统的中继器只是对所接收的信号进行放大，然后直接发送到另一个端口上连接的电缆上，主要用于扩展网络的物理连接范围；而网桥除了可以扩展网络的物理连接范围外，还可根据 MAC 地址进行寻址，实现数据交换，并且隔离了不同物理网段之间的碰撞（也就是隔离“冲突域”）。集线器和中继器都是物理层设备，而网桥属于二层设备。

这里涉及到两个非常重要的概念：“广播域”和“冲突域”。

1. 广播域

“广播域”是指一个广播数据包所能传播的范围，是针对三层 IP 网段的概念。在第 9 章中将介绍到，每一个自然网段或者一个子网都有一个特定的广播 IP 地址，这个广播 IP 地址就代表了发自该自然网段或者子网的 IP 数据包只能在本网段或子网内传播，不能逾越。也就是一个自然网段或者一个子网就是一个广播域。

对于中继器、集线器和二层交换机来说，因为它们只能实现在同一自然网段或者子网的设备连接，所以它们所连接的设备就同属于一个广播域。而对于三层设备（如三层交换机、路由器、网关）来说，因为它们的每个接口可以配置不同网段的 IP 地址，连接不同网段的设备，所以它们的每个三层接口所连接的都是一个独立的广播域。

2. 冲突域

“冲突域”是针对物理层的传输介质访问冲突而言的，即凡是同一时间可能形成介质访问冲突的范围就是一个冲突域。因为在同一条信道中，可能同时有两个方向，或者多路信号在传输，结果造成访问冲突。传输介质访问冲突控制是由数据链路层的 MAC 子层来实现的，具体将在下一章中介绍。

在物理层设备（如中继器、集成器）中，由于在设备内部的背板中只有一条传输信道（如图 6-41 所示），所以各端口连接的设备在进行通信时都是共享这一条信道，这样一来，整台设备所连接的网络都属于同一个冲突域，冲突概率比较大。

对于数据链路层设备（如网桥和二层交换机），由于在设备内部背板上有一个信道矩阵（如图 6-42 所示），所以不同端口所连接的设备间通信在背板上不存在冲突。但是不是这些二层设备就不存在介质访问冲突吗？答案当然是否定的。因为尽管在设备背板中不存在信道访问冲突，但是对于每一条传输介质来说，还可能因为同时存在双向的数据传输而造成访问冲突。所以这些二层设备的冲突域就明显比物理层设备的冲突域小，每一个设备端口所连接的那条电缆就是一个冲突域，冲突的概率小很多。

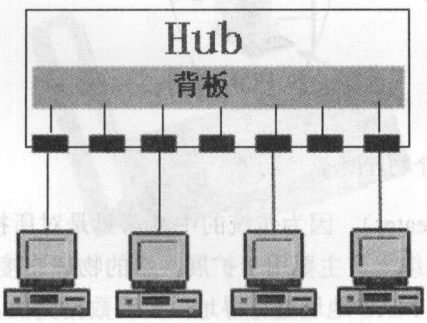


图 6-41　集线器端口与背板的总线连接示意图

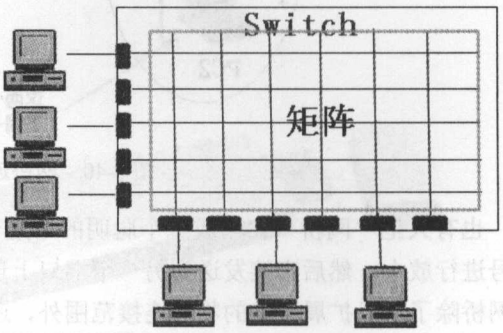


图 6-42　交换机端口与背板矩阵连接示意图

如图 6-43 所示是一个广播域和冲突域的示例。

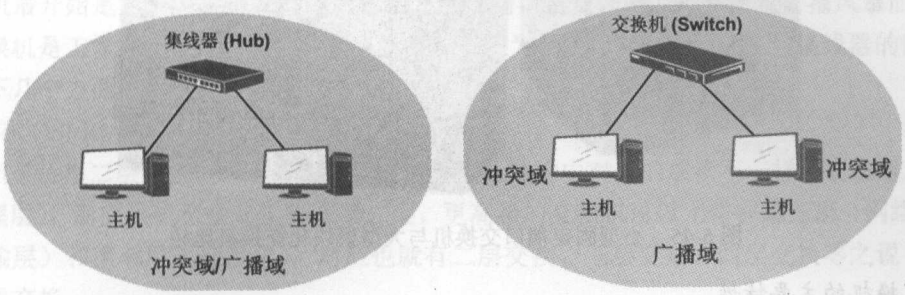


图 6-43 广播域和冲突域示例

至于网桥的数据交换原理与后面将要介绍的二层交换原理是一样的，因此这里不再赘述。

6.9.3 二层交换机

在计算机网络设备中还有一种是工作在数据链路层的，那就是二层交换机（其实三层或以上层次的交换机同样提供二层交换功能）。

交换机（Switch）可以说同时是集线器和网桥的升级换代产品，因为交换机既具有集线器的集中连接功能，同时又具有网桥的数据交换功能。所以可以说交换机是带有交换功能的集线器，或者说交换机是多端口的网桥。外形上，集线器与交换机产品也没什么太大区别，如图 6-44 所示，左图是一款集线器产品，右图是一款交换机产品。

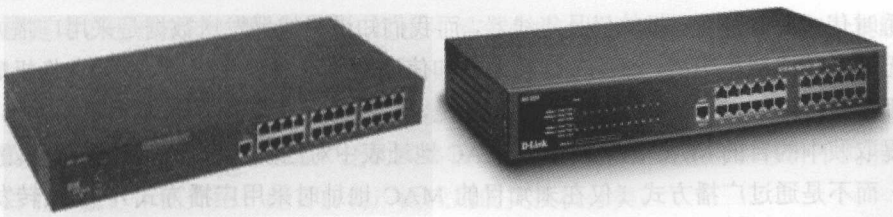


图 6-44 集线器与交换机的外观比较

从图中的对比可以看出，交换机与集线器的外观没什么本质区别，都是具有许多同类端口的网络设备。当然图中的对比仅能起到一般意义上的外观比较，实际上，因为交换机发展相当快，交换机的应用也向两个不同的方向发展，所以在外观上也有很大的区别。小的桌面交换机就像我们现在用的 Modem 一般大（集线器也有这样小的），而大的则采用模块化结构，机箱较大，而不是像图中显示的那样是一个长方形盒。如图 6-45 所示，左右两图分别代表小型固定端口的桌面型交换机和大型模块化交换机。



图 6-45　小型固定端口交换机与大型模块化交换机比较

1. 交换机的主要特性

交换机（此处特指二层交换机）是上节介绍的网桥的升级产品，同样是工作在网络体系结构中的第二层（数据链路层）。但它与网桥相比又具有一些特性，具体体现在以下几个方面：

（1）具有多个交换端口。

我们知道网桥通常只有两个交换端口，其设计目的主要就是用来连接两个距离超过单段网线传输限制的物理网段（当然也可以用来直接连接两台主机），所以它的应用受到比较多的限制。再加上当时用于主机和其他网络设备集中连接的设备仍是传输效率和信道利用率都非常低下的集线器，根本不适应于计算机网络的发展。

有了交换机后，一台交换机可以有多个端口，而且与网桥一样，不仅每个端口可以连接一个不同的物理网段（交换机上一个端口对应一个物理网段），还可以有大量的端口来集中连接主机，这时交换机就可以同时担当集线器和网桥的双重角色，而且在性能、扩展性能、交换性能等方面都有较大提高，大大促进了计算机网络的发展。

（2）数据转发效率更高。

在网桥时代，集中连接主机的仍是集线器，而我们知道集线器发送数据是采用广播方式，所以信道中的无效载荷比例相当高，造成数据转发率和信道利用率都非常低。而有了交换机后，因为大多数主机都是直接连接在交换机端口上，即使不是，也主要是连接在其他交换机端口上，数据的转发是通过提取帧中的目的 MAC 地址，根据 MAC 地址表中对应的出接口直接从连接目的主机的端口上发送，而不是通过广播方式（仅在未知目的 MAC 地址时采用广播方式），数据转发效率和信道利用率都得到大幅提高。

（3）更强的 MAC 地址自动学习功能。

我们知道，网桥通常只有两个端口，仅可以连接两个由集线器集中连接的物理网段，所以它的 MAC 地址自动学习功能仅限于它的两个端口与对应的物理网段的映射。这样就造成了一个网桥端口要与多个源主机 MAC 地址之间的映射，也就是一对多映射关系。而交换机上的端口多数是直接连接主机的，所以在映射表中基本上都是一个源主机 MAC 地址与一个交换端口间的一对一映射。一对一的映射查找起来明显比一对多的映射效率要高，所以交换机的数据转发效率要高于网桥。另外，交换机的缓存通常比网桥的要大，所以交换机中可以保存的 MAC 地址与端口映射表要多，更适用于较大网络。

2. 交换机与集线器的区别

交换机最开始是为了解决集线器共享传输介质、端口带宽过窄，容易产生广播风暴而产生的。最初的交换机是工作在体系结构的第二层，所以也称之为二层交换机。交换机与集线器的区别主要体现在以下几个方面：

（1）在 OSI/RM 中的工作层次不同。

交换机（特指二层交换机，下同）和集线器在体系结构中对应的层次不一样，集线器工作在第一层（物理层），而交换机至少是工作在第二层，更高级的交换机可以工作在第三层（网络层）、第四层（传输层）和第七层（应用层），对应也就有二层交换、三层交换、四层交换等之说了。本章仅介绍二层交换。

（2）数据传输方式不同。

集线器的数据传输方式是多次复制方式的广播传输，而交换机的数据传输是有目的的，数据只对目的节点发送，只是在自己的 MAC 地址表中找不到的情况下第一次使用以 FF-FF-FF-FF-FF-FF 作为目的 MAC 地址进行“泛洪”广播方式传输。所以，交换机在数据传输效率和信道利用率方面要远高于集线器，集线器更容易产生“广播风暴”。

（3）背板信道占用方式不同。

在带宽占用方面，集线器的所有端口是共享集线器背板中的一条信道带宽，而交换机的每个端口的收、发都有独享的背板信道带宽，是属于“交换”方式。这样一来更进一步使得交换机的数据传输效率和传输性能要远高于集线器。

（4）数据通信方式不同。

集线器因为是所有端口共享一条背板信道，所以只能采用半双工方式进行传输，同一时间，要么是接收数据，要么是发送数据。而交换机中各端口的信道是采用矩阵交换方式，可以同时进行数据交换，也就是可以进行全双工数据传输。这也使得交换机的数据通信效率要远高于集线器。

3. 交换机的分类

性能越强的设备应用越广，这也导致了该设备的技术不断发展，基于各种不同应用的设备类型也越多。交换机就是这样一种应用非常广泛的网络设备，它自最初出现至今，各种不同的交换机类型真是令人目不暇接。下面介绍一些目前仍广泛应用的交换机分类。

（1）根据网络类型划分。

根据交换机所应用的局域网类型可以将局域网交换机分为标准以太网交换机（10Mb/s 传输速率）、快速以太网交换机（100Mb/s 传输速率）、千兆以太网交换机（1000Mb/s 传输速率）、10 千兆以太网交换机（10000Mb/s 传输速率）等。

（2）按交换机结构划分。

如果按交换机结构来划分的话，交换机可分为：固定端口交换机和模块化交换机两种。固定端口顾名思义就是它所带有的端口是固定的，如果是 8 端口的，那就最多只能接 8 个设备，再不能添加；如果是 16 端口的，也就只能有 16 个端口，不能再扩展，依此类推。

目前这种固定端口的交换机基本上都属于较低档次的。模块化交换机就是交换机上除了有部分

固定的端口外，还可通过插入扩展模块来扩展端口数量、所支持的传输介质/网络协议/业务类型。如图 6-46 所示，左、右图分别是固定端口交换机和模块化交换机。

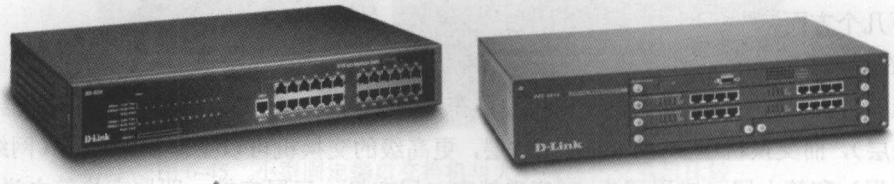


图 6-46　固定端口交换机和模块化交换机示例

（3）按是否支持网络管理功能划分。

如果按交换机是否支持网络管理功能，可以将交换机划分为“网管型”和“非网管型”两大类。“网管型交换机”是可以通过控制端口（Console）或 Web 界面进行配置与管理的，“非网管型交换机”则不能进行任何配置与管理，仅可按照出厂的默认设置进行工作，也就是通常所说的“傻瓜型交换机”。

从外观上，基本上可以判断一个交换机是否具有网管功能，因为网管型交换机都有一个 Console 控制端口，以前通常采用 RS-232 串口型的，现在主要采用 RJ-45 接口但会有一个 Console 字样标注的。如图 6-47 所示是一个带有 RS-232 控制端口的网管型交换机。

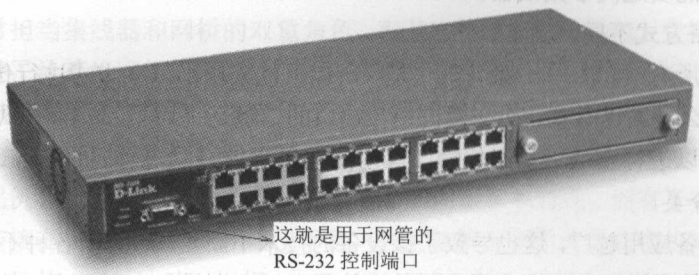


图 6-47　网管型交换机

6.9.4　二层交换原理

二层交换也就是在同一个局域网内部所进行的数据交换。在第 3 章介绍计算机网络体系结构时就已经讲到，尽管局域网体系结构中只有最低的物理层和数据链路层，但是也必须依靠整个计算机网络体系的各层来完成最终的用户网络应用目的，其中担当网络层以及各层角色的就是安装了网络操作系统的主机。

1. MAC 地址表

在整个二层交换过程中要用到以下两张表：

- 交换机上的 MAC 地址表：包括 MAC 地址、出接口（学习到对应 MAC 地址的入接口），

还可能包括 VLAN ID。

- 主机上的 ARP 映射表：IP 地址、MAC 地址、出接口（网卡）。

数据在计算机网络中转发路径的建立就是依靠以上两张表。但在二层交换中，在交换机设备上只建立二层的 MAC 地址表，ARP 映射表是仅在计算机主机上建立的；而在本书后面将要介绍的三层交换中在交换机上则需要同时建立二层的 MAC 地址表（用于在同一局域网内部的数据转发）和三层的 ARP 映射表（用于在不同 IP 网段之间的数据转发）。

MAC 地址表建立的是 MAC 地址与出接口、VLAN ID（仅在 VLAN 环境中存在）之间的映射关系。它可以使交换机在收到带有某个目的 MAC 地址的帧时，根据 MAC 地址表中的对应 MAC 地址表项选择本地设备中在同一个 VLAN 中的一个出接口进行数据转发。

MAC 地址表的建立主要有两种方式：一是由管理员根据网络环境手动创建；二是通过交换机设备的自动 MAC 地址学习功能来生成。自动 MAC 地址学习功能就是当交换机收到一个帧时，会把帧中的源 MAC 地址、VLAN ID 提取出来，然后再把接收这个帧的交换机端口作为与这个 MAC 地址对应的出接口形成一条 MAC 地址表项。这样下次收到一个帧中的目的 MAC 地址与该 MAC 地址相同时（也就是有其他设备向原来这台设备发送数据时），交换机会根据帧中的目的 MAC 地址在 MAC 地址表中查找对应的 MAC 地址表项，然后验证发送这个帧中的 VLAN ID 是否与这个 MAC 地址表项中的 VLAN ID 一致，一致则从该 MAC 地址表项中找到对应的出接口把数据发送出去，否则直接丢弃，因为在 VLAN 环境中仅允许同一 VLAN 中的用户直接进行二层通信。

2. 二层交换原理

前面已经提到，尽管在二层交换网络中二层交换机上仅会建立二层的 MAC 地址表，但还必须依靠在计算机上创建的 ARP 映射表，究其原因是因为在发送端没有获知目的端 MAC 地址时必须先依靠 ARP 广播报文来为它查找目的设备的 MAC 地址，否则无法在数据链路层上进行帧封装。当然，这也仅是在两端初次通信时才需要，因为后面通过 ARP 广播报文获知了对方 MAC 地址后就会在交换机上创建对应的 MAC 地址表，同时在计算机上也会创建对应的 ARP 映射表项，就可以直接从这些表中查找得到了。

在初次通信时，由于通信双方均不知道对方的 MAC 地址（一般只知道对方的 IP 地址），这里需要借助计算机上的 ARP 协议来解析 IP 地址所对应的 MAC 地址，具体流程如下（有关 ARP 协议报文格式和映射表建立方法将在第 8 章中介绍）：

（1）在不知道目的端 MAC 地址时，源端主机发送一个 ARP 广播包（目的 MAC 地址为广播 MAC 地址），以获取目的端的 MAC 地址。

（2）二层交换机在收到 ARP 广播帧后会根据帧中的源 MAC 地址、接收该帧的交换机端口和该端口所加入的 VLAN 建立基于源主机的 MAC 地址表项，然后从除了源端口外的其他所有端口上一级级地泛洪该 ARP 广播帧，直到直接连接目的主机的交换机。

（3）IP 地址配置为与 ARP 广播帧中目的 IP 地址相同的主机会接收此 ARP 广播帧并建立基于源主机的 ARP 映射表项，然后向源主机返回一个单播 ARP 响应包。

（4）二层交换机在收到目的主机发来的响应 ARP 帧后，根据帧中的源 MAC 地址、接收该响

应 ARP 帧的交换机端口和该端口所加入的 VLAN 建立基于目的主机的 MAC 地址表项，同时根据原来已建立好的基于源主机的 MAC 地址表项一级级地单播传送 ARP 响应帧，直到直接连接源主机的交换机。

（5）源主机收到来自目的主机的 ARP 响应包后建立基于目的主机的 ARP 映射表项。

此时源主机就已获知了目的主机的 MAC 地址，下次再要向该目的主机发送数据时可直接用所获得的目的主机 MAC 地址封装要传输的数据帧发送给交换机，此时交换机可直接利用原来已建立好的基于目的主机的 MAC 地址进行数据转发，最终实现源主机与目的主机的数据交换。

当然，如果主机发生了移动，连接到了其他交换机端口，当交换机收到主机发来的数据帧时会根据帧中的源 MAC 地址找到对应的 MAC 地址表项，然后进行更新。

6.10 课后自测题

为了巩固本章所学的内容，请认真做下面的自测题，参考答案见附录。

6.10.1 填空题

1．在局域网中数据链路层可分为两个子层，其中_____子层可用来解决介质争用，而与物理层介质访问无关的功能如_____、_____、_____等都放在_____子层。

2．数据链路层的主要功能包括_____、_____、_____和_____4 个方面。

3．数据链路层的服务包括_____、_____和_____3 种类型，其中_____包括_____、_____和_____3 个主要阶段，在 WLAN 通信中采用的是_____，在以太网通信中采用的是_____。

4．常见的帧同步方法有_____、_____、_____和_____。

5．BSC 协议头部所用的控制字符是_____，尾部所用的控制字符是_____；HDLC 协议需要在帧头和帧尾都用特定的比特串_____进行标识。

6．在数据链路层中差错控制功能分为_____和_____两个方面。前者主要有_____和_____两种方法，后者通常采用的方法包括_____、_____、_____和_____等几种。

7．“反馈检测法”要求_____，以此来检测对方是否正确接收了对应数据帧，同时还采用了_____和_____两项附加技术。

8．ARQ 差错控制法主要有_____和_____两种方案，前一方案规定_____，仅当接收方确认正确接收后再继续发送下一帧，其主要不足是_____；后一方案是指发送方可以_____，不用等前一帧被确认便可继续发送下一帧，当出现传输差错时可采用_____和_____两种策略，前者的主要不足是传输效率低。

9．海明码纠错的计算和纠错原理包括以下几个步骤：_____、_____、_____和_____，校验码必须在_____位置。正常情况下（即信息码在传输过程中不出现差错），如果采用奇校验，则各校验组的校验结果均应为_____；如果采用偶校验，则各校验组的校验结果均

应为_____。

10. 数据链路层的流量控制方案主要有_____和_____两种，主要用于面向字符的异步通信协议的流量控制方案是_____。

11. HDLC 提供了_____、_____和_____3 种基本操作方式，除此之外还有 3 种扩展操作方式，即_____、_____和_____。

12. 在 SDLC/HDLC 中，信号码的起始和结束是以_____标识的，在不进行帧传送的时刻，为了维持信道处于激活状态，发送端会_____。为了实现透明传输，当信号码中出现连续 5 个“1”时需要在后面插入一个_____。

13. PPP 帧的“控制”字段固定为_____。为了实现透明传输，使用了_____作为转义字符，将“信息”字段中出现的每一个 0x7E 字节转变成 2 字节序列_____，即_____。

6.10.2 选择题（可多选）

1. 以一个特殊字符代表一个帧的起始，并以一个专门的字段来标识帧字节数的帧同步方法是（ ）。

A. 字节计数法　　B. 字符填充首尾定界法

C. 违法编码法　　D. 比特填充首尾定界法

2. 在每个数据块的头部用一个或多个同步字符 SYN 来标记数据块的开始的帧同步方法是（ ）。

A. 字节计数法　　B. 字符填充首尾定界法

C. 违法编码法　　D. 比特填充首尾定界法

3. 在帧头和帧尾各插入一个特定的比特串来标记一个数据帧的起始与结束的帧同步方法是（ ）。

A. 字节计数法　　B. 字符填充首尾定界法

C. 违法编码法　　D. 比特填充首尾定界法

4. 曼彻斯特编码所采用的帧同步方法是（ ）。

A. 字节计数法　　B. 字符填充首尾定界法

C. 违法编码法　　D. 比特填充首尾定界法

5. 在 IEEE 802 局域网中采用的帧同步方法是（ ）。

A. 字节计数法　　B. 字符填充首尾定界法

C. 违法编码法　　D. 比特填充首尾定界法

6. 在数据链路层的差错控制方案中，（ ）是无须使用任何特殊代码进行错误检测的。

A. 反馈控制法　　B. 自动重发请求

C. 空闲重发请求　　D. 连续重发请求

7. 在反馈检测法发现不符时，由向接收端发送一个控制字符（ ）通知接收端删去出错数据。

A. DLE　　B. DEL　　C. SYN　　D. DCI

8. 反馈检测法进行差错控制主要应用于（　　）。

A. 面向比特的异步传输中　　B. 面向比特的同步传输中

C. 面向字符的异步传输中　　D. 面向字符的同步传输中

9. 发送窗口和接收窗口都大于 1 的差错控制方法是（　　）。

A. 空闲重发请求　　B. Go-Back-N

C. 选择重发　　D. 都不是

10. 数据链路层的流量控制实际上是对（　　）的控制。

A. 发送方数据流量　　B. 接收方数据流量

C. 接收、发送两方数据流量　　D. 链路上任意两节点间数据流量

11. XON/XOFF 流量控制方案发送的一对控制字符是由（　　）来发送的。

A. XON：发送方，XOFF：接收方　　B. XON：接收方，XOFF：发送方

C. XON、XOFF 都是由接收方　　D. XON、XOFF 都是由发送方

12. 下列差错控制方法中不需要使用冗余检错码的是（　　）。

A. 反馈检测法　　B. 空闲重发请求法

C. 连续重发请求法　　D. 自动重发请求法

13. 下面有关自动重发请求差错控制法的特点说法错误的是（　　）。

A. 需要携带 PCC 或 CRC 之类的检错码

B. 发送端需要一个缓存空间，接收端不需要

C. 发送端和接收端都需要一个缓存空间

D. 不需要重传整个数据帧回发送端

14. 下面有关选择重发策略差错控制方法的特点说法错误的是（　　）。

A. 当接收到错误帧时立即停止后续帧的接收，等待发送端重传错误的帧

B. 当接收到错误帧时仍可继续接收后续的帧，并提交给网络层

C. 当接收到错误帧时仍可继续接收后续的帧，并先放进缓存中，等待错误的帧重传到达后再一起提交给网络层

D. 发送端的重发表中的帧重发是按照 FIFO（先进先出）的机制进行排列的

15. 假设信息码一共是 12 位，如采用海明码进行纠错，则需要的校验码位数是（　　）。

A. 2　　B. 3　　C. 4　　D. 5

16. 透明传输是数据链路层的基本功能，它的目的是使（　　）。

A. 传输中的数据内容不受控制字符、标志符位的影响

B. 传输中的数据内容、格式及编码不受限制

C. 传输中的数据可以不管所采用的数据链路层协议

D. 传输中的数据对用户透明

17. 为了实现数据的透明传输，BSC 常用（　　）方法。

A. 比特填充首尾标志法　　B. 字节计数法

C．字符填充首尾定界符法　　D．违法编码法

18．下面的数据链路层协议中是面向字符的有（　　）。

A．BSC　　B．DDCMP　　C．SDLC

D．PPP　　E．HDLC

19．下面的数据链路层协议中是面向比特的有（　　）。

A．ADCCP　　B．DDCMP　　C．SDLC

D．PPP　　E．HDLC

20．BSC 协议的数据块格式中，靠近报头之前的那个控制字符是（　　）。

A．STX　　B．SYN　　C．SOH　　D．ETX

21．在 BSC 协议的数据块格式中，块校验字符为（　　）。

A．SYN　　B．BCC

C．CRC　　D．BSC

22．在 BSC 协议中块校验字符 BCC 的校验范围为（　　）。

A．自 SOH 始，至 ETX 止　　B．自 STX 始，至 ETX 止

C．自 STX 始，至 ETB 止　　D．自 STX 始，到 ETX 或 ETB 止

23．在 HDLC 的操作方式中，主站点控制着整个链路的操作的是（　　）。

A．ARM　　B．NRM

C．ABM　　D．SARM

24．为了达到数据的透明性，使用字符填充的首尾定界符法采用（　　）。

A．0 比特插入法　　B．转义字符填充法

C．增加冗余位　　D．以上都不是

25．在 BSC 协议中，EOT 字符的作用是（　　）。

A．用作轮询和选择监控报文

B．用以标志报文的交换结束，并在两点间拆除逻辑链路

C．用作对选择监控信息的响应

D．用以对先前所发数据块是否正确接收作出响应

26．HDLC 是（　　）。

A．面向字符的同步协议　　B．面向比特的同步协议

C．面向字计数的同步协议　　D．异步协议

27．在 HDLC 中，用来封装用户数据的帧是（　　）。

A．I 帧　　B．S 帧

C．U 帧　　D．UI 帧

28．下列 HDLC S 帧中只用于选择重发差错控制中的是（　　）。

A．S 字段为 00 的帧　　B．S 字段为 01 的帧

C．S 字段为 10 的帧　　D．S 字段为 11 的帧

29. 下面关于 SLIP 协议的说法不正确的是（　　）。

A. 不能自动分配 IP 地址　　B. 仅可在 IP 网络中使用

C. 无检错能力　　D. 不能实现透明传输

30. 网桥或交换机的 MAC 地址表中保存的是（　　）。

A. 主机 MAC 地址

B. 主机 MAC 地址与所连网桥或交换机端口之间的映射

C. 主机 MAC 地址与其 IP 地址之间的映射

D. 主机 MAC 地址和所属 VLAN 的 VLAN ID 与所连网桥或交换机端口之间的映射

31. 下面关于 PPP 协议说法正确的是（　　）。

A. 上层协议仅支持 IP 协议

B. 可只在运行 PPP 协议的链路一端接口配置 IP 地址

C. 运行 PPP 协议的链路两端接口 IP 地址可以不在同一网段

D. PPP 协议的帧分为 I 帧、U 帧、S 帧 3 类

32. 下面关于 PPP 协议帧格式说法正确的是（　　）。

A. PPP 协议的“标志”字段固定为 11111110

B.“地址”字段是源端的 MAC 地址

C. PPP 帧中的“地址”字段与 HDLC 帧中的“地址”字段一样固定为 11111111，无实际意义

D. 目的端在收到 PPP 帧后必须进行源端确认

33. 关于 PPP 协议组件说法不正确的是（　　）。

A. LCP 负责 PPP 链路的协商

B. 在链路建立过程中，NCP 在 PPP 认证之后进行

C. 在链路拆除过程中，NCP 要先于 LCP 关闭

D. PPP 链路必须进行认证，且通过后才能建立

34. 下面关于 PPP 认证说法错误的是（　　）

A. PAP 认证仅支持单向认证，CHAP 同时支持单向和双向认证

B. PAP 认证仅需要两次握手，CHAP 认证需要三次握手

C. PAP 认证是由客户端发起的，CHAP 认证是由服务器端发起的

D. PAP 认证只支持明文认证，CHAP 认证支持 MD5 加密认证

35. 下面关于二层交换说法错误的是（　　）。

A. 二层交换仅适用于二层交换网络

B. 二层交换不需要用到 ARP 映射表

C. 二层交换中交换机上仅会建立 MAC 地址表

D. 二层交换中 MAC 地址的寻址也要依靠主机发送的 ARP 请求报文

6.10.3 计算题

1. 假设要发送的信号码为 11101100101，分别计算采用 PCC 奇、偶校验方式后传输的信号码。

2. 假设要发送的信号码为 100101110，且 CRC 生成多项式为 $g(x)= x^5 + x^4 +x+1$，计算采用 CRC 校验方式后传输的信号码。

3. 假设要发送的信号码为 11000101，分别计算采用海明码奇、偶校验纠错方式后传输的信号码。

7

介质访问控制子层

介质访问控制子层（MAC 子层）是局域网体系结构中划分的子层，对于广播型网络（如以太局域网、WLAN）是非常重要的，因为它担负了两方面的主要职责：一是如何在局域网内进行 MAC 寻址（也就是找到目的节点）；二是如何解决多路通信中介质访问冲突的现象。

MAC 寻址的原理比较简单，就是根据帧中的“目的 MAC 地址”来查找交换机上已建立的 MAC 地址表项，找到对应的出接口，把数据帧一个个节点地发送出去。具体参见上一章介绍的二层交换原理。但是要注意，数据帧在每个三层节点处都要重新封装，也就是每经过一个三层设备，帧头中的“源 MAC 地址”和“目的 MAC 地址”都会变的，但在同一 IP 网段的局域网内部，帧头的 MAC 地址信息是始终不变的。

本章主要针对 MAC 子层在解决介质访问冲突方面的功能及实现原理介绍了最初的 CSMA 协议以及后来在标准以太网、快速以太网中应用的 CSMA/CD 协议的介质访问控制原理（在全双工的千兆和万兆以太网中不需要这些协议），然后着重介绍了各种主要的以太网规范、帧格式，以及各种以太网标准所包括的以太网规范和体系结构。

本章重点与难点：

- CSMA 和 CSMA/CD 协议的介质访问控制原理
- 主要的以太网标准帧格式
- LLC 帧头部和 SNAP 头部格式
- MAC 帧格式
- 千兆和万兆以太网规范及体系结构

7.1 MAC 子层基础

“MAC 子层”是局域网体系结构中数据链路层的一个子层，它有两个主要作用：一是用来寻址（这里是 MAC 地址），也就是寻找目的节点；二是用来解决网络中多个用户争抢共享物理介质或者共享信道的现象。如在总线型局域网中，所有或者多个用户计算机连接在一条总线上，不同用户计算机间的通信肯定会存在介质争用现象。



大家千万不要认为仅有完全的总线型拓扑结构网络才有介质争用现象发生，其实只要网络中存在“总线”部分，也就是只要网络中存在可能有多个用户通过同一条介质访问同一个节点的情况，就存在介质争用。即使在最简单的星型结构单元以太网中也会存在介质争用现象，因为每个用户计算机与交换机的连接都是通过一条电缆进行的，网络中可能有多个其他用户要同时访问该用户。在企业网络中主要使用的树型结构以太网中，这种介质争用现象就更普遍了，因为不同交换机之间的连接也通常是通过一条级连（Uplink）电缆进行的，这样两个交换机之间用户的访问就都必须通过这一条级连电缆，介质争用就更不可避免了。

在 Internet 的 TCP/IP 协议体系结构中没有 MAC 子层，因为在 TCP/IP 协议体系结构中主要描述的是广域网中的“通信子网”部分，在这部分主要设备就是各种广域网路由器（一般没有传统的以太网交换机），而在这些广域网路由器之间的连接基本上都是点对点的连接，不存在寻址，也不存在介质争用现象。

7.1.1 两种信道类型

说到“MAC 子层”就不得不提到“信道”这个概念，因为 MAC 子层的主要功能之一就是为了解决不同用户通信信道争用问题。

信道可分为“点对点信道”和“广播信道”两大类。“点对点信道”是在由两个没有经过任何中间设备（也就是两个设备的接口是“背靠背”连接的）的节点间构成的点对点链路上运行点对点类型数据链路层协议（如 PPP 和 PPPoE 等）而形成的信道，如图 7-1 所示。

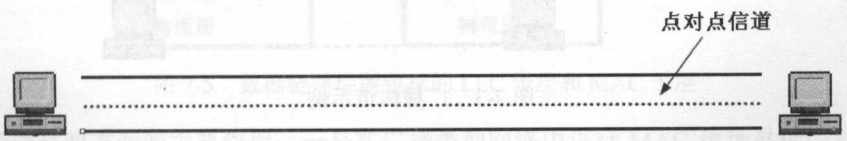


图 7-1 点对点信道示例

因为“点对点链路”两端的节点只与对方节点进行连接，所以不存在寻址问题。如图 7-2 所示的网络中，计算机 A 可以（注意，不是必须，因为具体还要看所封装的数据链路层协议）与计算

机 B、计算机 C、计算机 E 建立点对点连接，但却不能与计算机 D 建立点对点连接，因为计算机 A 与计算机 D 之间建立连接时都必须经过一个中间设备（计算机 B，或者计算机 C，或者计算机 E），就不属于点对点连接了。计算机 E 却可以与其中的任何一台计算机建立点对点连接。注意，本示例中每台计算机与另一台计算机的连接都是通过一块单独网卡进行的，这里的点对点链路是在两台计算机上安装的两块计算机网卡之间建立的。

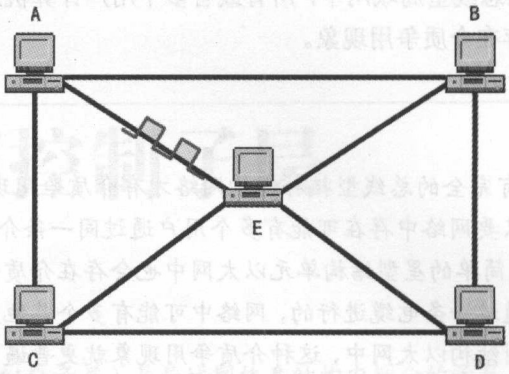


图 7-2　点对点连接示例

“广播信道”表现为一个信道被多条链路所共享，也就是存在物理介质共享的情况，如图 7-3 所示各计算机均连接在同一条总线上。在广播信道中，一个节点发送的数据可以同时被多个节点接收到，对应的链路就是“广播链路”。当然这里也要求链路上所运行的数据链路协议是支持广播传输方式的，最典型的就是以太网协议和 WLAN 协议。在这些广播网络中，一个用户发送的一个广播包可以通过交换机广播到本局域网中的所有节点上。如图 7-4 所示是一个总线型拓扑结构网络中数据在广播信道中同时向多个节点传输。

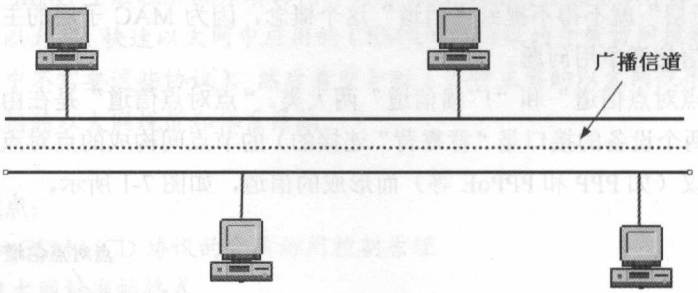


图 7-3　广播信道示例

从以上分析可以看出，MAC 子层仅在广播型网络（如以太局域网、WLAN，以及其他无线网络）中有用，而对于点对点型网络来说就没有存在的意义了，因为在点对点网络中是不需要寻址的，也不存在一般意义上的信道争用问题（即使存在，也基本上可以通过各种信道复用技术来解决）。

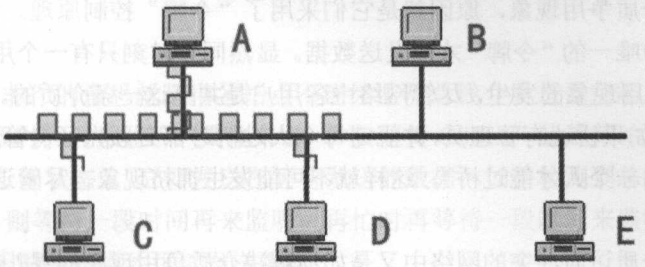


图 7-4 广播数据传输示例

7.1.2 MAC 子层概述

局域网（包括令牌网、以太局域网和 WLAN 等）中与接入的各种传输介质相关的问题都放在 MAC 子层来解决，而且 MAC 子层还负责在物理层的基础上实现无差错的通信。

因为计算机网络体系结构中的“数据链路层”分成了“LLC 子层”和“MAC 子层”（如图 7-5 所示），所以数据链路层有两种不同的数据帧：LLC 帧和 MAC 帧。不过，我们通常所说的“帧”是指 MAC 帧而不是 LLC 帧，因为 MAC 子层位于 LLC 子层之下。从网络层来的数据包，进入到 LLC 子层后加上 LLC 子层的协议头就形成了 LLC 帧，到达 MAC 子层后再要加上 MAC 子层的协议头和协议尾，又要进行重新封装，最终形成 MAC 帧传输到物理层。对于物理层来说，最终的数据链路层帧就是指 MAC 帧。

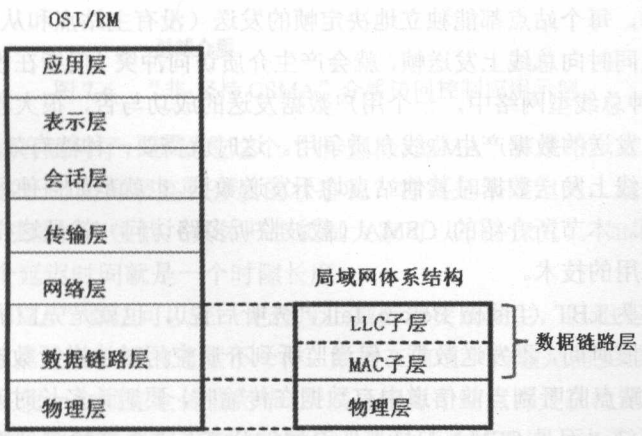


图 7-5 数据链路层所包括的 LLC 子层和 MAC 子层

MAC 子层有两方面的主要作用：一是在广播类型网络中进行 MAC 地址寻址；二是进行介质访问冲突控制。MAC 地址寻址是根据帧中的目的 MAC 地址进行数据帧的转发，介质访问冲突控制就是为了避免冲突的发生，但不同网络所采用的冲突解决方法是不一样的。

在第 2 章介绍 IBM 令牌环网络和令牌总线网络时说到，在这两种看似都是采用共享介质的令

牌网络中却不存在介质争用现象，原因就是它们采用了“令牌”控制原理。就是在这两种网络中，用户必须得到网络中唯一的“令牌”才能发送数据。显然同一时刻只有一个用户节点可以得到这个令牌，避免了介质争用现象的发生，尽管网络中各用户是共享这一条介质的。就像许多人要从两个方向过一座独木桥，如果桥边有管理员，并规定每个人过桥时都必须先征得管理员同意才能过桥（也就是等一人过完后另一个人才能过桥），这样就不可能发生拥挤现象，尽管这条桥很窄，一次仅允许一人通过。

那么其他存在介质访问冲突的网络中又是如何解决介质争用现象的呢？在广域网中，通常是采用各种对应的信道复用技术来实现的，如我们在第 5 章中介绍的 FDM 信道复用技术就是把一条高带宽的信道划分成带宽较小的多个信道，这样就可以同时进行多路通信了，就像在一条公路上划分多个小车道，可以同时通过多部小车一样。TDM 信道复用技术是通过把不同路通信的数据包分配在不同时隙进行传输来实现信道的复用。

在常见的以太局域网和 WLAN 无线局域网中介质争用解决方案主要有：CSMA（Carrier Sense Multiple Access，载波监听多路访问）、CSMA/CD（Carrier Sense Multiple Access/Collision Detection，载波监听多路访问/冲突检测）和 CSMA/CA（Carrier Sense Multiple Access/Collision Avoidance，载波监听多路访问/冲突避免）协议，后两者是前者的改进版本，这里先介绍在以太局域网中使用的 CSMA 和 CSMA/CD，适用于 WLAN 的 CSMA/CA 将在 7.9.4 节中介绍。

7.2 CSMA 介质访问控制原理

在总线型网络中，每个站点都能独立地决定帧的发送（没有主站点和从站点之分），很显然，如果两个或多个站点同时向总线上发送帧，就会产生介质访问冲突（仅指在没有采用信道复用的情况下）。因此，在这种总线型网络中，一个用户数据发送的成功与否，很大程度上取决于发送数据时是否会与其他用户发送的数据产生总线介质争用。这时就需要一种能有效避免冲突发生的技术，确保每个站点在向总线上发送数据时其他站点均不发送数据，也就是如何使各个站点能尽快地检测到总线介质是否空闲。本节所介绍的 CSMA（载波监听多路访问）就是这样一种能比较有效解决总线型网络中介质争用的技术。

CSMA 技术也称为 LBT（Listen Before Talk，先听后说），也就是先监听要访问的介质，当发现介质忙时先避让一段时间，不发送数据，仅当监听到介质空闲时才进行数据发送。在这里就涉及到一个问题，就是在站点监听到当前信道中有数据在传输时，要避让多长时间才再次监听，这就是 CSMA 技术的退避算法。而且 CSMA 只能监听介质的状态，并不能检测到冲突的发生，更不能真正有效地避免冲突的发生，所以这只是一种初级的介质访问冲突解决文案，本章后面将要介绍的 CSMA/CD 和 CSMA/CA 是它的改进版本。

CSMA 可以采用的退避算法有几种，也可以算是对应类型的 CSMA，即：非-坚持 CSMA（no persistent CSMA）、1-坚持 CSMA（1-persistentes CSMA）、p-坚持 CSMA（p-persistant CSMA）。下面是这些算法的具体诠释。

7.2.1 非-坚持算法

“非-坚持”中“非”的意思就是指各站点不连续监听总线介质是否空闲，即在发现介质忙时先停止监听，等过一段时间再来监听。具体来讲，“非-坚持 CSMA”就是某站点发现总线介质上没有数据在传输时（处于闲状态），立即发送在等待发送的数据；如果检测到总线介质上有数据在传输时（处于忙状态），则等待一段时间再来监听，再忙时再等待一段时间来监听，依此类推，直到发现总线介质处于空闲时本站点才可以发送数据。

如图 7-6 所示是一个“非-坚持 CSMA”介质访问控制示例。示例中某站点准备要发送一个数据帧 Fi。它第一次检测时发现总线是处于忙状态，于是隔了一段时间 t1 后再来监听，还发现总线介质是处于忙状态，于是它再隔一段时间 t2 来监听，终于在第三次监听时发现总线介质是处于闲状态，于是这个站点立即把 Fi 帧发送出去。

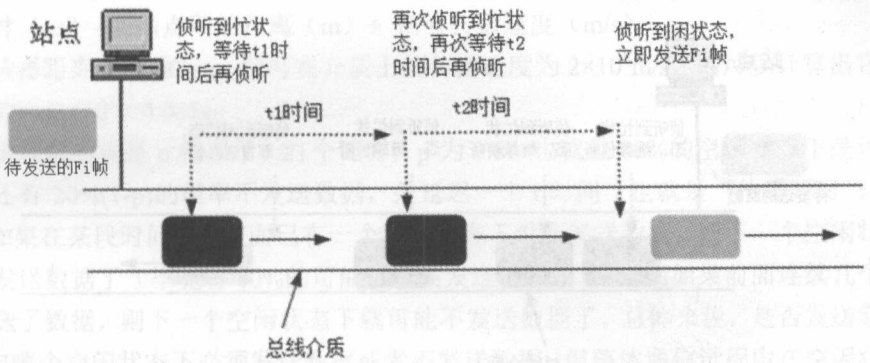


图 7-6 “非-坚持 CSMA”介质访问控制应用示例

这时大家可能已经有疑问了，那就是这个“非-坚持 CSMA”算法中，在上次监听发现介质处于忙状态时需要等待多长时间再来监听介质的状态呢？答案是不固定的，这个等待的时间是根据“非-坚持 CSMA”算法随机产生的（具体产生机制大家可以不必深究）。如果采用 TDM（时分复用）传输方式，则这个延迟时间就是一个时隙长度。

虽然“非-坚持 CSMA”算法可以在一定程度上减少冲突的发生，但还是不能完全消除冲突的发生，因为毕竟每个站点发送数据的速度不一样，数据在信道中传输是需要时间的，而且等待的时间也不一定就可以满足使一个帧完全传输完毕，所以尽管每个站点都是在检测到介质空闲时才发送数据，但仍可能存在前一个帧还在信道中传输时后一个站点的帧就追上来了，发生冲突。

另外，“非-坚持 CSMA”算法还存在一个明显的缺点，这就是一旦监听到介质忙就马上延迟一个随机时间再重新监听，但很可能在再次监听之前信道就已经空闲了。也就是“非-坚持 CSMA”不能把信道刚变成空闲的那一时刻精准地找出来，影响了信道利用率的提高。所以这种算法主要适用于小型总线或者树型拓扑结构网络中，不适用于像现在的大型树型结构以太网中。为了克服这一缺点，可以采用下面介绍的两种“坚持 CSMA”。

7.2.2 1-坚持算法

“1-坚持”中的“1”有两层含义：一是指发现总线介质忙时一直持续、不间断监听，直到发现介质处于闲状态；二是在监听到介质处于空闲状态后一定（也就是100%）发送数据。前面介绍的“非-坚持 CSMA”算法是在发现介质忙后即随机等待一个延时，然后继续监听，发现介质处于闲状态后立即发送数据；而此处的“1-坚持 CSMA”算法中是在发现介质忙时不等待、继续监听，一旦发现空闲即立即发送数据；在数据传送过程中发生冲突时放弃当前的数据传送任务，在等待一个延时后再继续监听。

如图 7-7 所示是一个“1-坚持 CSMA”介质访问控制示例。示例中某站点准备要发送一个数据帧 Fi。它第一次检测时发现总线是处于忙状态，于是继续监听（不等待），第二次监听还是发现总线介质是处于忙状态，于是它再继续监听，一直到第 n 次监听时才发现总线介质是处于闲状态，于是这个站点立即把 Fi 帧发送出去。

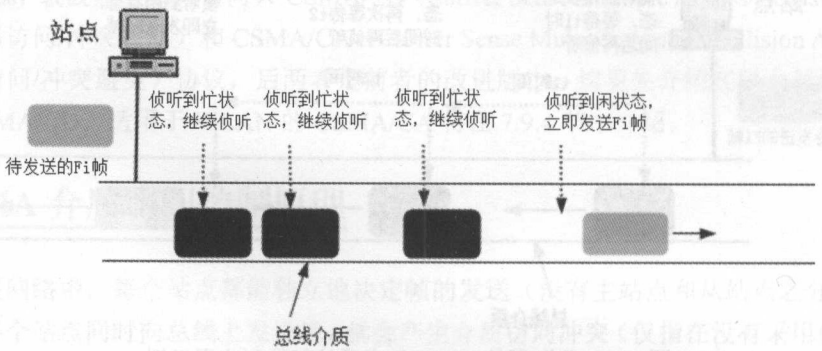


图 7-7 “1-坚持 CSMA”介质访问控制应用示例

很明显，这种算法相对前面介绍的“非-坚持”算法来说的优点就是提高了介质的利用率，每个站点都在抓住一切有利时机发送数据，因为在没有发生冲突时无需等待一个随机延迟就立即进行继续监听，不会出现介质处于空闲状态仍没有站点发送数据的情况。

但“1-坚持”算法仍有两个致命的弱点：一是在有多个站点发送数据的情况下，这种毫不等待的算法也就使得冲突时常发生，原因就是前面所说的，可能在网络中同时有多个站点在同一时间检测到介质空闲（因为中间没有一个延迟，也就是一直在监听介质状态）而立即进行了数据发送，所以更容易发生冲突；二是这种算法在发现介质忙时一直监听，占用了大量网络和设备资源。所以“1-坚持 CSMA”算法也仅适用于小型总线型或者树型拓扑结构网络，不适用于像现在的大型树型结构以太网中。

7.2.3 p-坚持算法

既然前面介绍的两种算法都存在明显的不足，自然就会有人继续后面的开发，于是就生产了新

的“p-坚持 CSMA”退避算法。其实“p-坚持 CSMA”算法是前面介绍的“非-坚持 CSMA”和“1-坚持 CSMA”这两种算法的一种折衷算法，取两者的长处而尽量克服了两者的不足。

理解“p-坚持 CSMA”退避算法的关键就是其中的“p”，是指监听到介质空闲时发送数据的概率（小于 1，也就是小于 100%），就是指站点在发现介质空闲时可以立即发送数据的概率为 p，也就是不一定（不是 100%）发送数据，还有(1-p)概率是不发送数据的。其目的就是为了避免与其他站点发生冲突。

具体来讲，“p-坚持 CSMA”算法的具体介质访问控制方法与前面介绍的“1-坚持 CSMA”类似，也是在监听到介质处于忙状态时持续监听，当监听到介质处于空闲状态时，此站点不一定会马上发送数据，而是根据概率 p（这个 p 值是算法事先确定好的）来选择发送数据，而在另一个(1-p)概率的时候，即使介质处于空闲状态，也会延迟一段时间 t（这个 t 是指端到端的“传播延时”）后再重新监听介质状态。

上面所说的“传播延时”是有计算公式的，具体如下：

传播延时（μs）=两站点间的距离（m）÷ 信号传播速度（m/s）

如已知传播距离为 1000km，信号在介质上的传播速度为 2×10^8m/s，则可以计算出它的传播延时为 $1000\times1000/(2\times10^8)=0.005$s。

这里最难理解的就是 p 概率了。打个比方，p 为 0.8，也就是 80%的空闲状态下是可以立即发送数据的，还有 20%(1-p)的概率不发送数据，是延迟一个 t 时间（也就是“传播延时”）再来监听介质状态。如果在某段时间中，前面已有一个空闲状态下没有发送数据，则下一个空闲状态下就很可能会立即发送数据了（毕竟有 80%的可能性是要发送数据的），否则如果前面连续几个空闲状态下都立即发送了数据，则下一个空闲状态下就可能不发送数据了。总体来说，是否发送数据是随机的，没有固定哪个空闲状态下必须发送数据或者不发送数据，但整体通信过程中在空闲状态下发送数据的概率都是在 p 以内。

如图 7-8 所示是一个“p-坚持 CSMA”介质访问控制示例。示例中某站点准备要发送一个数据帧 Fi。它第一次检测时发现总线是处于忙状态，于是继续监听（不等待），第二次监听还是发现总线介质是处于忙状态，于是它再继续监听，第三次监听时发现总线介质是处于闲状态，但根据概率计算出现在不能发送数据，于是等待一个 t 时间后再来监听介质状态，到第 4 次监听时发现介质还是处于空闲状态，根据概率计算得出马上可以发送数据了，于是把 Fi 帧发送出去。

从以上分析可以看出，“p-坚持”退避算法是一种既能像“非-坚持 CSMA”算法那样减少冲突，又能像“1-坚持 CSMA”算法那样减少介质空闲时间的折衷方案，也就是综合了前面所介绍的两种算法的优点，以实现优缺点互补。

但是不管是以上哪种 CSMA 算法，它们均只能监听出介质的状态，对在数据传输过程中可能出现的冲突现象仍不能有效地检测，所以直接采用 CSMA 协议来进行介质访问冲突控制的仅是那些小型总线型网络，因为在比较大的网络中发生冲突的概率仍然非常高。正因为如此，在现在广泛应用的以太局域网中就不是直接采用 CSMA 介质访问冲突解决方案，而是采用下节将要介绍的同时具有介质状态监听和冲突检测功能的 CSMA/CD 协议。

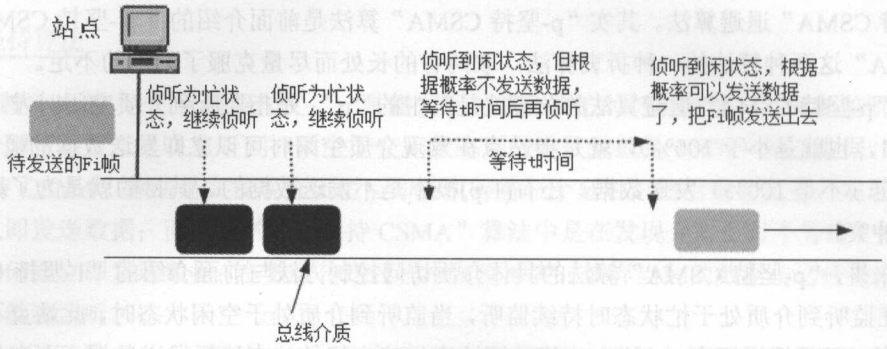


图 7-8 “p-坚持 CSMA”介质访问控制应用示例

7.3 CSMA/CD 介质访问控制原理

在上节介绍的 CSMA 介质访问控制中，由于信道传播时延的存在，即使通信双方的站点都没有监听到载波信号，在发送数据时仍可能会发生冲突，因为他们可能会在检测到介质空闲时同时发送数据，致使冲突发生。尽管 CSMA 可以发现介质空闲状态，但它并没有先知的冲突检测和阻止功能。而本节要介绍的 CSMA/CD（Carrier Sense Multiple Access with Collision Detection，载波监听多路访问/冲突检测）技术却具有冲突检测和阻止功能，它是 CSMA 技术的改进版本，是由 IEEE 802.3 标准定义的。

CSMA/CD 是标准以太网、快速以太网和千兆以太网中统一采用的介质争用处理协议（但在万兆以太网中，由于采用的是全双工通信，所以不再采用这一协议）。CSMA/CD 的最大亮点就是它在监听到有冲突发生时（同时有多路数据发送时介质上的电平强度肯定不一样）可以立即中止数据帧的发送，快速地终止被破坏的帧可以节省时间和带宽；并且发送一个阻塞信号，以强化冲突，使其他站点更容易检测到有冲突发生，不再同时发送数据了，可避免更多的帧发生冲突。

7.3.1 CSMA/CD 原理综述

CSMA/CD 介质访问控制原理包含 4 个处理内容：监听、发送、检测、冲突处理，可以用以下几句话来概括：

- 先听后说（“听”是指“监听”，“说”是指发送数据，下同），边听边说。
- 一旦冲突，立即停说。
- 等待时机，然后再说。

具体解释如下（可参见图 7-9 所示的示例）：

（1）当一个站点想要发送数据的时候，它首先要检测总线介质上是否有其他站点正在传输，即监听介质是否空闲（也就是前面所说的“先听”）。

（2）如果信道忙，则继续监听，直到监听到介质状态为空闲；如果监听到介质状态为空闲，

站点就准备好要发送的数据（也就是前面所说的“后说”）。

（3）在发送数据的同时，站点继续监听总线介质（也就是前面所说的“边听边说”），确信没有其他站点在同时传输数据才继续传输数据。因为有可能两个或多个站点都同时检测到介质空闲，然后几乎在同一时刻开始传输数据。如果两个或多个站点同时发送数据，就会产生冲突。若无冲突则继续发送直到发完全部数据。

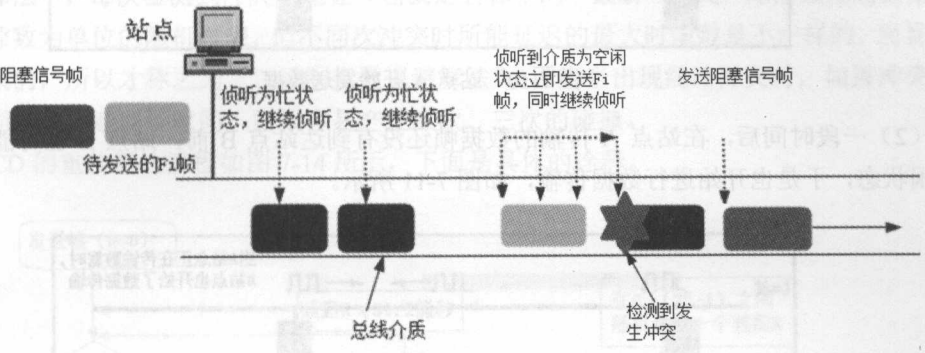


图 7-9 CSMA/CD 介质访问控制应用示例

（4）若检测到有冲突，则立即停止发送数据（也就是前面所说的“一旦冲突，立即停说”），同时发送一个用于加强冲突的 JAM（阻塞）信号，以便使网络上所有站点都知道网上发生了冲突，不再接收原来的帧，转而接收这个阻塞帧。然后本站点等待一个预定的随机时间（也就是前面所说的“等待时机”），且在总线为空闲时再重新发送数据（也就是前面所说的“然后再说”）。

有人将 CSMA/CD 的工作原理形象地比喻成很多人在一间黑屋子中举行讨论会，参加会议的人都是只能听到其他人的声音却看不到人。会议中规定，每个人在说话前必须先倾听，只有等会场安静下来后他才能够发言。这时，可将人们在发言前进行的监听（以确定是否已有人在发言）的动作比喻为 CSMA/CD 的“载波监听”；将在会场安静的情况下每人都有平等机会讲话比喻为 CSMA/CD 的“多路访问”。如果有两人或两人以上同时说话，大家就无法听清其中任何一人的发言，这种情况称为发生“冲突”；发言人在发言过程中要及时发现是否发生冲突，这个动作称为“冲突检测”；如果发言人发现冲突已经发生，这时他需要停止讲话，然后随机后退延迟，再次重复上述过程，直至讲话成功。如果失败次数太多，他也许就放弃了这次发言的想法。

7.3.2 冲突检测原理

为了确保所有站点在发送数据的站点发送完数据前开始接收数据，以太网规定了一个最小帧（这个最小帧的有效载荷不能少于 46 个字节），最小帧的大小与网络分布的距离、传输介质类型和到达目的节点前所使用的中继器数量等有关。综合这些因素专家们定义了一个公认值——以太网时隙，为 10Mb/s 速率下传输 512 比特所用的时间，即 51.2μs。

当有两个或多个正在传输数据的站点检测到它们发送的数据发生了冲突时，它们都会通过发送

一个阻塞帧（它是一个 32 位全为 1 的帧）来进行响应。下面以一个示例来描述 CSMA/CD 的冲突检测原理。

（1）一开始（t=0 时），站点 A 在介质空闲时发送一个帧，如图 7-10 所示。

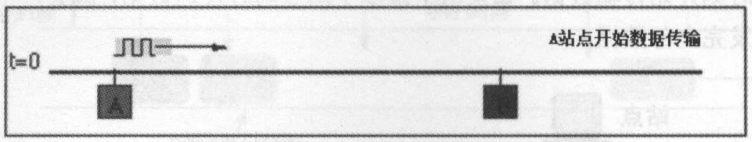


图 7-10　站点 A 开始发送数据

（2）一段时间后，在站点 A 传输的数据帧还没有到达站点 B 前，站点 B 也可能检测到介质为空闲状态，于是也开始进行数据传输，如图 7-11 所示。

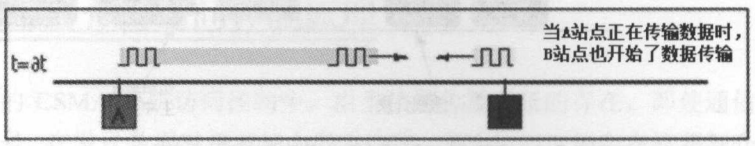


图 7-11　站点 B 也开始数据传输

（3）从 A 站点发送数据开始，过一段时间后（相当于站点 A 到达站点 B 的传播延时 tp），站点 B 检测到了来自站点 A 的数据，检测到发生了冲突，立即中止原来的数据传输，发送一个 32 位的阻塞信号帧。但是站点 A 此时仍还不知道站点 B 也在发送数据，仍不知道有冲突发生，如图 7-12 所示。

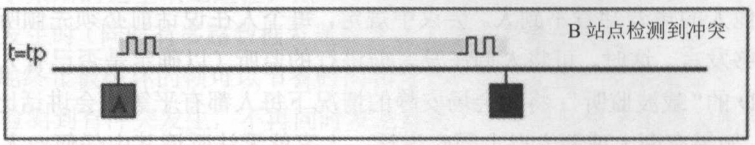


图 7-12　站点 B 检测到冲突

（4）又过了相当于一个往返传播时间（也就是自站点 A 发送数据开始过了两个传播延时 2tp）后，从站点 B 发来的阻塞信号帧也到达了站点 A，A 也知道发生冲突了，停止数据发送，如图 7-13 所示。此时站点 B 会停止阻塞信号帧的发送，而站点 A 又会发送一个阻塞信号帧。最后，A 和 B 都停止了数据发送，介质回归空闲状态。

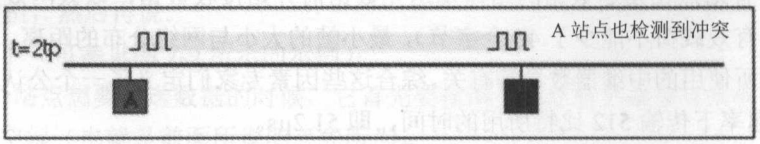


图 7-13　站点 A 也检测到冲突

7.3.3 冲突避让原理

CSMA/CD 除了可以检测到冲突的发生外，还可以在冲突发生时进行随机延迟，以尽量避免再次发生冲突。这时又涉及到一个“避让延迟”问题，也就是当检测到冲突时到底要延迟多长时间再重传。这就是 CSMA/CD 的“截断二进制指数退避算法”（Truncated Binary Exponential Backoff）。在这种退避算法中，每次检测到冲突时避让（也就是名称中的“截断”含义，即停止发送数据）的时间是以时隙数为单位的随机时间，但不同次冲突时所能延迟的最大时隙数是不一样的，而且是呈指数递增关系的，所以才称之为“二进制指数退避算法”。因为当出现线路冲突时，如果冲突的各站点都采用同样的退避间隔时间，则很容易产生二次、三次的碰撞。

CSMA/CD 的重传避让流程如图 7-14 所示，下面是具体的诠释。

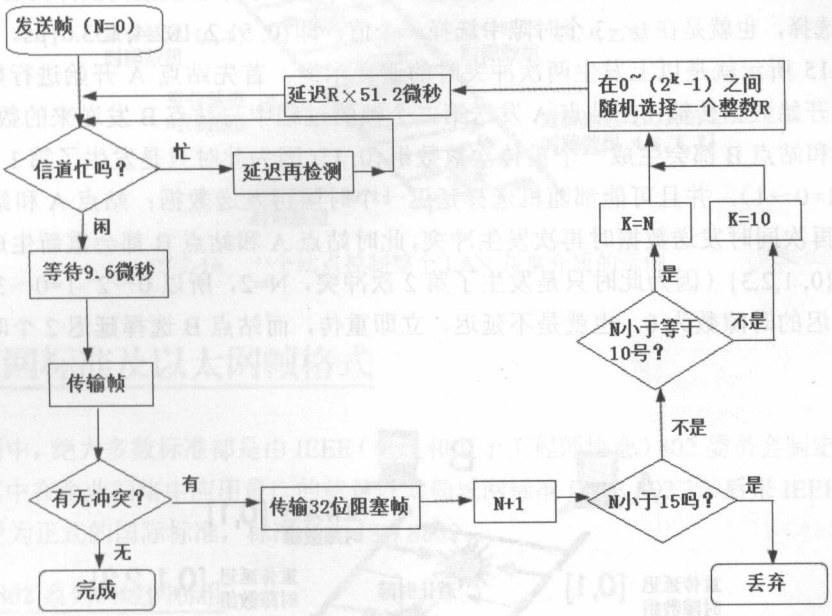


图 7-14 CSMA/CD 避让原理示意图

（1）站点会初始化当前帧的重传次数 N 为 0 并开始监听介质状态。如果介质处于忙状态，则等待，在等到介质处于空闲状态后再等待一个帧间发送间隙的时间（IFG，为 9.6μs），以使网络中所有接收站点有时间准备好接收下一个帧。

（2）站点开始按次序传送帧，同时继续监听介质状态。

（3）如果传到某个帧时监听到发生了冲突，则该站点立即停止传输，开始冲突处理流程，发送一个用于加强冲突的阻塞信号，并把该帧的重传次数 N 递增 1。

（4）到了这里，就要对 N 值进行判断了，因为不同的重传次数可以延迟的时隙数不一样。因为如果所有站点在发生冲突后都立即重传，则肯定又会发生另一个冲突，所以需要一个正确的流程

来确保一个同时重传的情况很少发生，在以太网中所采用的方案就是使用一个随机的避让时间，使每个站点选择一个随机数来乘以时隙长度（51.2μs），在企图重传前必须等待这个随机的时长。

发生冲突时，CSMA/CD 可以计算出重传的次数（图中的 N），并规定当 N（重传次数）达到 15 时，该帧会被直接丢弃；当 N 小于 15 时，再判断 N 是否≤10，如果不是，则把 K 的值置为 10，否则 K=N，然后根据公式 R×51.2μs 来计算要延迟的时间。其中 R 是在 0～2^{k-1} 之间的一个随机整数，而 K=N，N 为传输次数（K=N≤10）。也就是说每个帧传输的次数是有限制的，那就是 16，即 N=15，超过这个值则会直接丢弃这个帧，放弃再次重传并生成一个日志。通常在一个负载不是很重的网络中是不会丢弃帧的，因为很难有重传 15 次后仍不能传输成功的。

每一次重传，站点都会构建一组可以选择的避让时隙数组{0, 1, 2, 3, 4, 5, ..., L}，其中 L=2^k-1，且 K≤10。每次重传时所选择的避让时隙个数 R 就是从以上数组中选择的。例如，在发生两次冲突后，N=2，所以 K=2，得到的数组就是{0, 1, 2, 3}（2^2-1），表示在发生两次冲突后所需避让的时隙数有 4 个选择，也就是在 0～3 个时隙中选择一个值，即{0, 51.2, 102.4, 153.6}μs。

如图 7-15 所示就是以上发生两次冲突时的避让示例。首先站点 A 开始进行数据传输，然后站点 B 也开始数据传输；在站点 A 发送第二个帧的过程中与站点 B 发送来的数据发生冲突，此时站点 A 和站点 B 都会生成一个重传次数数组{0, 1}（因为此时只是发生了第 1 次冲突，N=1，所以 0～2^1-1=0～1），并且可能都随机选择延迟一个时隙再发送数据；站点 A 和站点 B 在延迟一个时隙后再次同时发送数据时再次发生冲突，此时站点 A 和站点 B 都会重新生成一个重传延迟时隙数组{0, 1,2,3}（因为此时只是发生了第 2 次冲突，N=2，所以 0～2^2-1=0～3），但此时站点 A 选择延迟的时隙数为 0，也就是不延迟，立即重传，而站点 B 选择延迟 2 个时隙数后再重传数据。

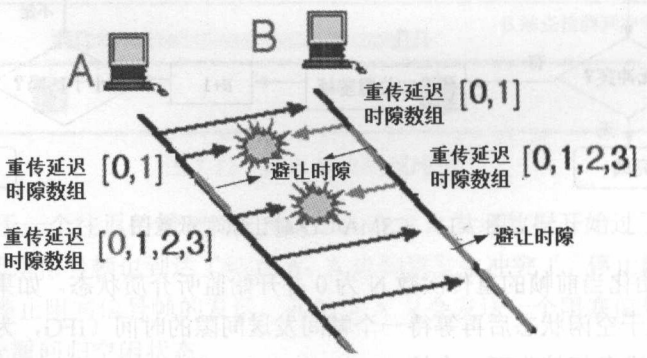


图 7-15　发生两次冲突时的避让示例

同理，在 3 次冲突后，N=3，可选择的避让时隙数组为{0, 1, 2, 3, 4, 5, 6, 7}（2^3-1），也就是可以在等待以上其中 8 个之一可选时隙数长度后再重传数据。4 次冲突之后，N=4，可选择的避让时隙数组为{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}（2^4-1），也就是可以在等待以上其中 16 个之一可选时隙数长度后再重传数据。

7.3.4 CSMA/CD 的不足

使用 CSMA/CD 介质访问控制的一个缺点就是，当 LAN 中互联的每台计算机都只有少量数据需要传输时，网络中每个站点对介质的共享都几乎是公平的；但是如果一个站点需要发送大量的数据时（如一个站点担当高质量视频源的情况下），这时就可能出现一个站点长时间控制整个 LAN 的情形。如图 7-16 所示，计算机 A 控制着计算机 B。最初，两台计算机都有数据在传输。A 先进行数据传输，然后 A 和 B 同时试图传输，发生了冲突。此时 B 选择了一个比 A 更大一些的等待时隙数来进行重传。A 开始重传，这时有一个短时间的暂停，然后 A 和 B 都试图恢复传输。

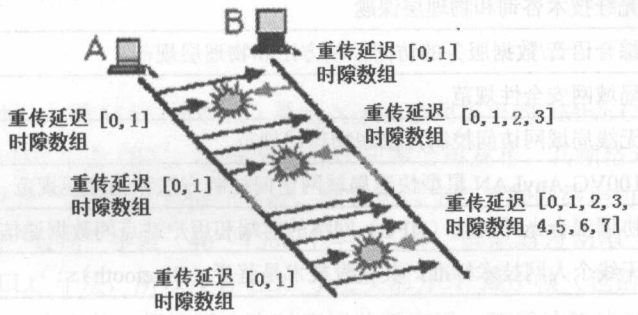


图 7-16 一个站点控制整个 LAN 共享介质的示例

7.4 局域网标准及以太网帧格式

在局域网中，绝大多数标准都是由 IEEE（电气和电子工程师协会）802 委员会制定的 IEEE 802 系列标准，其中在企业网络中应用最广的就是以太局域网标准 IEEE 802.3。后来 IEEE 802 系列标准被 ISO 接受为正式的国际标准，标准名为 ISO 8802。

7.4.1 IEEE 802 系列局域网标准

1985 年 IEEE 公布了应用于局域网的 IEEE 802 标准文本，同年被美国国家标准局（ANSI）采纳作为美国国家标准。后来，国际标准化组织（ISO）经过讨论，建议将 802 标准定为局域网国际标准。目前，IEEE 802 主要包括的标准如表 7-1 所示，在许多标准中又有许多规范，就像 IEEE 802.3 标准中就包括了许多不同的以太网规范，这些在本章后面再具体介绍。这些 802 分委员会标准之间以及各自与 OSI 参考模型之间的关系如图 7-17 所示。

表 7-1 主要 IEEE 802 标准

标准	说明
IEEE 802.1	局域网概述、局域网体系结构、网络管理和网络互联

续表

标准	说明
IEEE 802.2	逻辑链路控制 LLC
IEEE 802.3	CSMA/CD 总线介质访问控制子层与物理层规范
IEEE 802.4	令牌总线（Token Bus）介质访问控制子层与物理层规范
IEEE 802.5	令牌环（Token Ring）介质访问控制子层与物理层规范
IEEE 802.6	城域网（MAN）介质访问控制子层与物理层规范
IEEE 802.7	宽带技术咨询和物理层课题与建议实施
IEEE 802.8	光纤技术咨询和物理层课题
IEEE 802.9	综合语音/数据服务的访问控制方法和物理层规范
IEEE 802.10	局域网安全性规范
IEEE 802.11	无线局域网访问控制方法和物理层规范
IEEE 802.12	100VG-AnyLAN 星型快速局域网访问控制方法和物理层规范
IEEE 802.14	协调混合光纤同轴（HFC）网络的前端和用户站点间数据通信的协议
IEEE 802.15	无线个人网技术标准，其代表技术是蓝牙（Bluetooth）

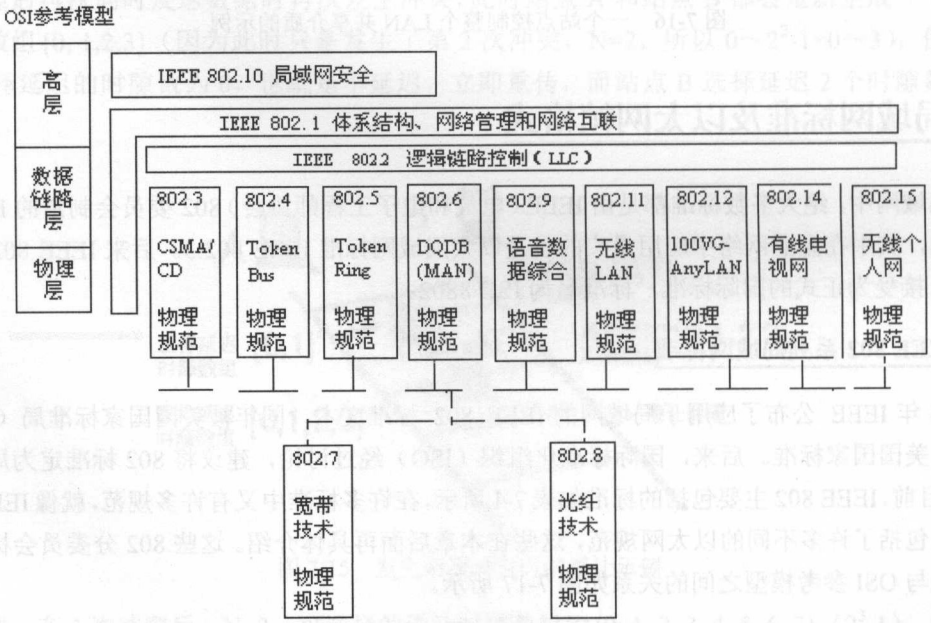


图 7-17　IEEE 各标准之间以及与 OSI 参考模型之间的关系

IEEE 802.1、802.2、802.3 系列接入规范都将在本章后面有专门介绍，WLAN IEEE 802.11 系列接入规范已在第 5 章中介绍过了，其他局域网规范的具体功能大家可以参考其他资料。

7.4.2 以太网帧格式综述

以太网（Enthernet）技术最初是在 1972 年由施乐公司的帕洛阿尔托研究中心（PARC）开发的，后来再联合数字设备公司（Digital Equipment）和英特尔（Intel）公司于 1982 年公布了其改进版本，也就是现在仍广泛使用的 Ethernet II。

自 1985 年后，随着以太网技术和应用的发展，IEEE（电子与电气工程师协会）成立了专门的 802 委员会，公布了一系列的局域网标准，其中 802.3 针对以太网，802.4 针对令牌总线网络，802.5 针对令牌环网络等（参见表 7-1），这些网络由共同的 IEEE 802.2 标准来定义其逻辑链路控制（LLC）子层，但 MAC 子层是由不同的标准来定义的。下面具体介绍截至目前为止所出现的一些主要以太网 MAC 标准。

（1）Ethernet I。

这是最原始的一种以太网 MAC 帧格式，是由 Xerox 公司提出的 3Mb/s CSMA/CD 以太网标准的封装格式，后来在 1980 年由 DEC、Intel 和 Xerox 三家公司发布。其帧格式如图 7-18 所示，其中最关键的一个字段就是“类型”（Type）字段，使它可以支持多种网络层协议包，如 IP、IPX、Apple Talk 协议等。除“数据”字段（46～1500 个字节）外，其他均为 MAC 子层的协议头和协议尾（此时并没有划分 LLC 子层，所以没有 LLC 子层头部分），最小帧（包括帧头和帧尾）长度为 64 个字节，最长为 1518 个字节。但该版本当时应用并不广泛，随后被新的 Ethernet II 版本所全面取代。

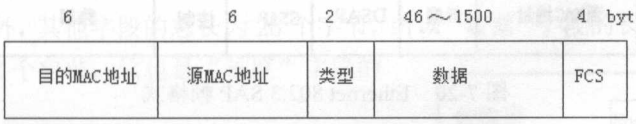


图 7-18 Ethernet I 帧格式

（2）Ethernet II。

这是由 DEC、Intel 和 Xerox 三家公司在 1982 年发布的，即 DIX2.0 的以太网帧格式。它主要更改了 Ethernet I 的电气特性和物理接口，在帧格式上并没有变化，参见图 7-18。Ethernet II 出现后迅速取代 Ethernet I 成为以太网事实标准。

（3）Ethernet 802.3 raw。

这是 Novell 公司在 1983 年公布的专用以太网标准帧格式，因为它仅支持 Novell 网络中使用的 IPX 协议。该格式以当时尚未正式发布（所以加了一个 raw，就是“原始”的意思）的 IEEE 802.3 标准为基础，具体的帧格式如图 7-19 所示。

与图 7-18 中的 Ethernet I/II 标准对比可以看出有两处不同：一是 Ethernet 802.3 raw 帧中多了一个 2 个字节的 0xFFFF，用于表示它是 Novell 协议以太网帧；二是原来在 Ethernet I/II 标准中的“类型”字段变成了“长度”字段，因为 Ethernet 802.3 raw 标准仅支持 IPX 这一种上层协议，不需要

“类型”字段来定义。“长度”字段用来以字节为单位标识帧中“数据”字段的大小。

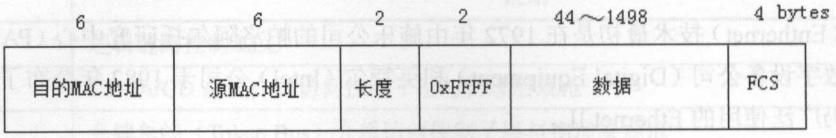


图 7-19　Ethernet 802.3 raw 帧格式

由于总的以太网帧最小和最大长度是不会变的（仍分别为 64 个字节和 1518 个字节），现在多了 2 个字节的 0xFFFF，所以“数据”字段中的最小和最大长度也相应减小 2 个字节，为 44～1498 个字节。但是当两年后 IEEE 正式发布 802.3 标准时情况发生了变化，因为 IEEE 在 802.3 帧头中又加入了 802.2 LLC 头，这使得 Novell 的 RAW 802.3 格式与正式版的 IEEE 802.3 标准互不兼容。

（4）IEEE Ethernet 802.3 SAP。

这是 IEEE 在 1985 年公布的 Ethernet 802.3 的 SAP 版本以太网帧格式，是 IEEE 发布的第一个以太网帧格式版本，其帧格式如图 7-20 所示。SAP 是服务访问点的意思，表示添加了 LLC 帧头部，包含目的服务访问点（DSAP）、源服务访问点（SSAP）和控制（Crontrol）这 3 个字段，具体将在下节介绍。

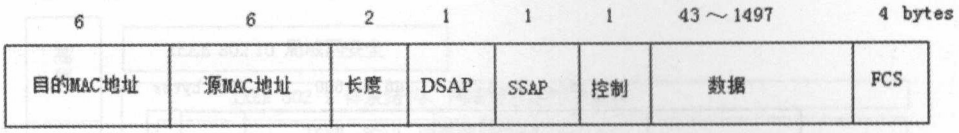


图 7-20　Ethernet 802.3 SAP 帧格式

另外，在 Ethernet 802.3 SAP 帧中将 Ethernet II 帧头的“类型”（Type）字段替换为帧“长度”（Length）字段，因为该标准仅支持 IP 协议，所以不需要“类型”字段来标识不同的上层协议。这样一来，Ethernet 802.3 SAP 帧也仅支持 IP 协议的 SAP。“长度”字段用来标识帧中“数据”字段的大小。

又因为在帧中新添加了来自 LLC 子层的 DSAP、SSAP 和 Crontrol 这 3 个各占 1 个字节的字段，所以真正的“数据”字段的取值范围也要相应调整为 43～1497 个字节。但 Ethernet 802.3 SAP 以太网帧格式版本其实只是一个过渡版本，因为 IEEE 在后面很快又进行了更新，相继发布了 802.3/802.2 LLC 和 802.3/802.2 SNAP 版本。

（5）802.3/802.2 LLC。

这是 IEEE 1997 年正式发布的 802.3 标准，由 Ethernet 802.3 SAP 版本发展而来，且帧格式也基本一样，但它又把 Ethernet 802.3 SAP 中的“长度”字段改为“类型”字段（如图 7-21 所示），因为它又需要支持多种上层协议，包括 STP、ARP、IP、IPX、AppleTalk 等，对应就有多个上层协议的 SAP，如 IP 为 0x06，IPX 为 0xE0，OSI 为 0xFE，STP 为 0x42。

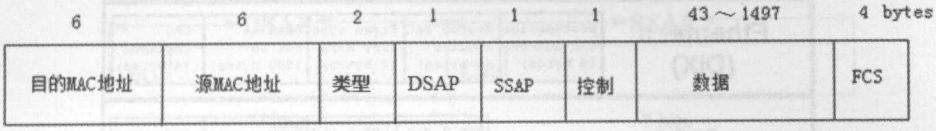


图 7-21　802.3/802.2 LLC 帧格式

（6）802.3/802.2 SNAP。

这是 IEEE 为保证在 802.3/802.2 LLC 标准上支持多种上层协议的同时更好地支持 IP 协议而于 1998 年发布的扩展以太网帧标准。SNAP 的全称是 SubNetwork Access Protocol，中文意思就是“子网访问协议”，是一种可以传输多种协议包的网络访问协议。

与 802.3/802.2 LLC 帧格式一样，802.3/802.2 SNAP 也带有 LLC 头，但是扩展了 LLC 属性：一是新添加了一个 2 个字节的“长度”字段，用于标识“数据”字段的大小；二是新添加了一个 3 个字节的 OUI ID（组织唯一标识，通常是全为 0）字段，用于代表发布所选上层协议的组织，如图 7-22 所示。

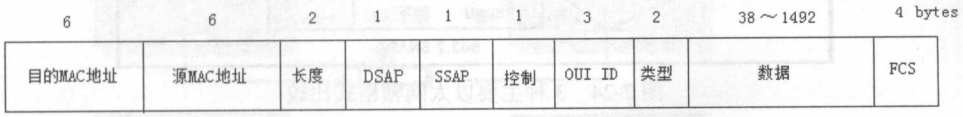


图 7-22　802.3/802.2 SNAP 帧格式

同样因为总的以太网帧最小和最大帧长度是不变的（仍分别为 64 个字节和 1518 个字节），现在除了“数据”字段外，其他字段的总长为 26 个字节，所以“数据”字段的长度范围为(64-26)=38～(1518-26) =38～1492 个字节，这也是“长度”字段的取值范围。

在以上 6 种以太网帧格式中，Ethernet I、Ethernet II 和 Ethernet 802.3 raw 这 3 个以太网帧仅包含 MAC 帧部分，而 802.3 SAP、802.2 LLC 和 802.3/802.2 SNAP 标准包括了 MAC 帧和 LLC 帧两部分。来自网络层的数据包到达数据链路层后要经过 LLC 和 MAC 这两个子层的两次先后封装。MAC 帧是把整个 LLC 帧当成 Data（数据）字段，然后再加上自己的 MAC 帧头和帧尾就构成了最终的 MAC 帧，也就是我们通常所说的以太网帧，整个以太网帧的封装流程如图 7-23 所示。

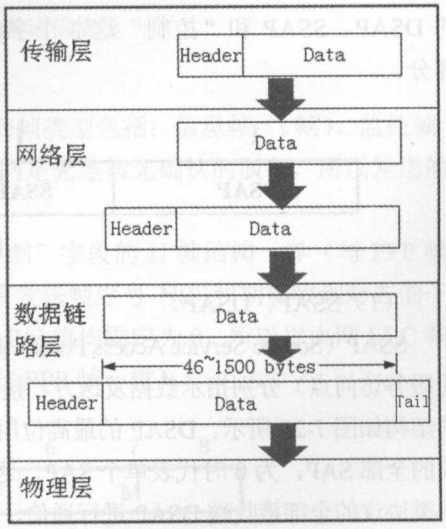


图 7-23　以太网帧的各级封装方式示意图

在以上这 6 种以太网帧格式中，当前主流应用的是 Ethernet II、802.3/802.2 LLC 和 802.3/802.2 SNAP 这 3 种，其格式对比如图 7-24 所示，各具体字段将在本节后面介绍。

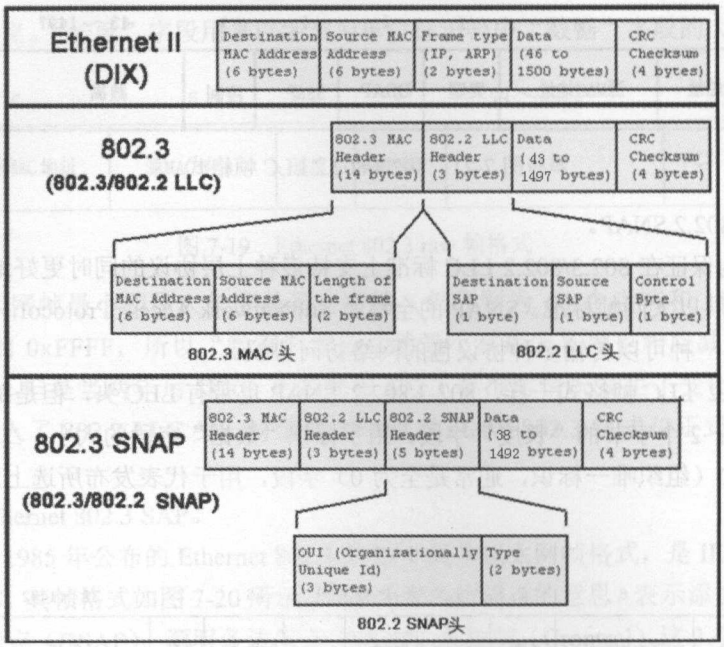


图 7-24　3 种主要以太网帧格式比较

7.4.3　LLC 帧头部格式

在上节已经介绍到，在 802.3 SAP、802.2 LLC 和 802.3/802.2 SNAP 这 3 种标准的 MAC 帧中是包括了 LLC 头部的，当然另外还有来自上层的数据部分，如图 7-25 所示是 LCC 帧格式。前面的 DSAP、SSAP 和“控制”这 3 个字段是属于 LCC 头部，“数据”字段就代表来自上层的数据部分。

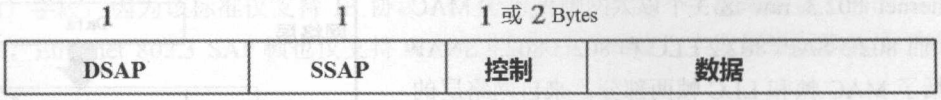


图 7-25　LLC 帧格式

（1）SSAP、DSAP。

SSAP（Source Service Access Point，源服务访问点）和 DSAP（Destination Service Access Point，目的服务访问点）分别指示数据发送方、接收方 LLC 子层的 SAP，占 1 个字节（8 位）。DSAP 和 SSAP 的结构如图 7-26 所示。DSAP 的最高位用来表示目的服务访问点的类型，为 1 时代表一组或者同类协议的全部 SAP，为 0 时代表单个 SAP。这样就可以实现一个发送端的 SSAP 可以与一个或者一组或者同类协议的全部接收端 DSAP 进行通信，如图 7-27 所示。而 SSAP 的最高位用来表示帧类型，为 1 时表示是响应帧，为 0 时表示是命令帧，在以太网协议中两者均固定为 0xAA。

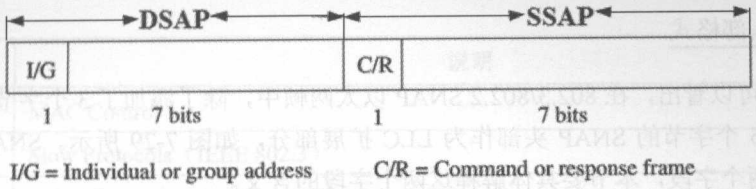


图 7-26 DSAP 和 SSAP 结构

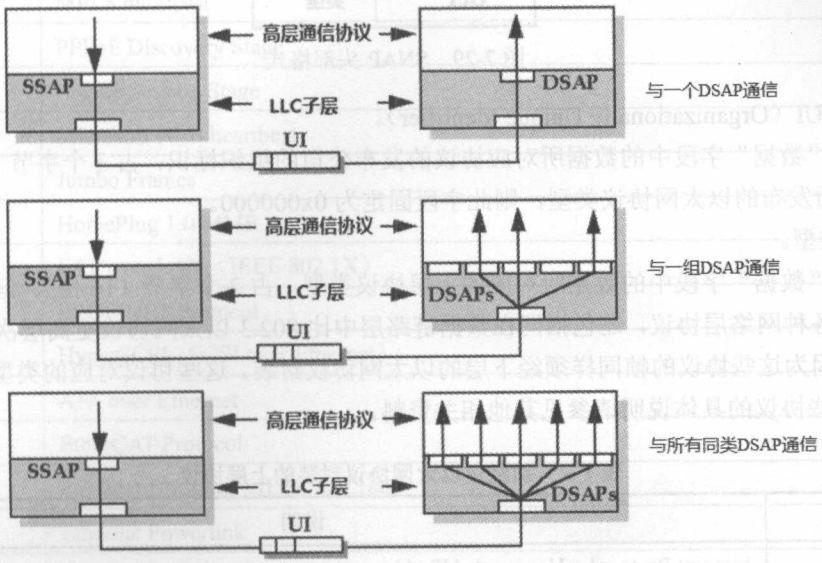


图 7-27 SSAP 与 DSAP 的 3 种通信情形

（2）控制。

“控制”字段用来指示 LLC 帧的类型，数据链路层的帧类型包括：信息帧（I 帧）、监控帧（S 帧）和无编号帧（U 帧），占 1 个字节（8 位）。因为以太网是无连接无确认的服务，所以发送的帧是无编号信息（UI）帧。

在以太网帧中，“控制”字段结构与 HDLC 帧中“控制”字段的 U 帧结构一样（与 PPP 帧中的“控制”字段结构也一样），如图 7-28 所示。因为没有发送帧序号 N(S)和期望收到对方的下一个帧的序号 N(R)对应的比特位，其 5 个 M 位和第 5 位的 P/F 值均固定为 0，所以以太网 LLC 帧中的“控制”字段的值是固定的，为十六进制的 0x03（也与 PPP 帧一样）。

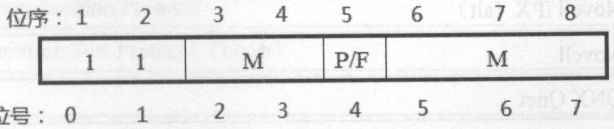


图 7-28 UI 帧“控制”字段结构

7.4.4 SNAP 头部格式

从图 7-24 可以看出，在 802.3/802.2 SNAP 以太网帧中，除了添加了 3 个字节的 LLC 头部外，还另外添加了 5 个字节的 SNAP 头部作为 LLC 扩展部分，如图 7-29 所示。SNAP 头部包括 OUI 和“类型”这两个字段，本节要具体解释这两个字段的含义。

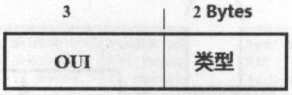


图 7-29　SNAP 头部格式

（1）OUI（Organizationally Unique Identifier）。

指示帧“数据”字段中的数据所对应协议的发布公司的组织标识，占 3 个字节（24 位）。如果是 IEEE 公司发布的以太网协议类型，则此字段固定为 0x000000。

（2）类型。

指示帧“数据”字段中的数据所对应的上层协议类型，占 2 个字节（16 位）。但这里“上层协议”不仅指各种网络层协议，还包括同在数据链路层中比 802.3 以太网协议更高层次的其他数据链路层协议，因为这些协议的帧同样须经下层的以太网协议封装。这些协议对应的类型值如表 7-2 所示。有关这些协议的具体说明请参见其他相关资料。

表 7-2　须经过以太网协议封装的上层协议

协议	说明
0x0800	Internet Protocol，Version 4（IPv4）
0x0806	Address Resolution Protocol（ARP）
0x0842	Wake-on-LAN Magic Packet
0x1337	SYN-3 heartbeat protocol（SYNdog）
0x22F3	IETF TRILL Protocol
0x6003	DECnet Phase IV
0x8035	Reverse Address Resolution Protocol（RARP）
0x809B	AppleTalk（Ethertalk）
0x80F3	AppleTalk Address Resolution Protocol（AARP）
0x8100	VLAN-tagged frame（IEEE 802.1Q）
0x8137	Novell IPX（alt）
0x8138	Novell
0x8204	QNX Qnet
0x86DD	Internet Protocol，Version 6（IPv6）

续表

协议	说明
0x8808	MAC Control
0x8809	Slow Protocols（IEEE 802.3）
0x8819	CobraNet
0x8847	MPLS unicast
0x8848	MPLS multicast
0x8863	PPPoE Discovery Stage
0x8864	PPPoE Session Stage
0x886F	Microsoft NLB heartbeat
0x8870	Jumbo Frames
0x887B	HomePlug 1.0 MME
0x888E	EAP over LAN（IEEE 802.1X）
0x8892	PROFINET Protocol
0x889A	HyperSCSI（SCSI over Ethernet）
0x88A2	ATA over Ethernet
0x88A4	EtherCAT Protocol
0x88A8	Provider Bridging（IEEE 802.1ad）
0x88AB	Ethernet Powerlink
0x88CC	LLDP
0x88CD	sercos III
0x88D8	Circuit Emulation Services over Ethernet（MEF-8）
0x88E1	HomePlug AV MME
0x88E3	Media Redundancy Protocol（IEC62439-2）
0x88E5	MAC security（IEEE 802.1AE）
0x88F7	Precision Time Protocol（IEEE 1588）
0x8902	IEEE 802.1ag Connectivity Fault Management（CFM）Protocol/ITU-T Recommendation Y.1731（OAM）
0x8906	Fibre Channel over Ethernet
0x8914	FCoE Initialization Protocol
0x9000	Configuration Test Protocol（Loop）
0x9100	Q-in-Q

7.4.5 MAC帧格式

MAC帧，也通常被认为就是数据链路层帧，是在MAC子层实体间交换的协议数据单元(PDU)。根据图7-24可以把所有以太网MAC帧的格式统一表示为如图7-30所示，前面两节介绍的LLC帧头和SNAP头部与上层来的数据报一起均封装在“数据”字段中。

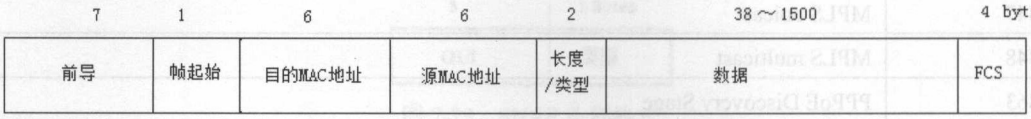


图7-30 MAC帧格式

（1）前导。

前导（Preamble）字段占7个字节，由1和0交互构成（如10101010……），用于使PLS子层电路与收到的帧达到时钟同步。

（2）帧起始。

帧起始（Start-of-Frame Delimiter，SFD）字段占1个字节，前6位也是由1和0交互构成，最后两位是连续的1，即10101011，表示一个帧的开始。前导码的作用是使接收端能根据1和0交互的比特模式迅速实现比特同步，当检测到连续两位“1”（即读到帧起始定界符字段SFD最末两位）时，便将后续的信息递交给MAC子层。

【说明】在以上两个字段中，在早期的Intel和Xerox公司开发的以太网标准中是把SFD字段并入到了Pre字段中，所以那时的MAC帧格式中没有SFD字段。只有后面由IEEE发布的以太网标准中才出现了SFD字段。但Pre和SFD这两个字段只是用来提醒接收端新的一帧到来了，并不计入MAC帧大小中。

（3）目的MAC地址/源MAC地址。

目的MAC地址（Destination Address，DA）和源MAC地址（Source Addresses，SA）字段各占6个字节，分别用于标识接收站点的MAC地址和发送站点的MAC地址。

MAC地址又分单播MAC地址、组播MAC地址和广播MAC地址3种。最高位为“0”表示单播MAC地址，该地址仅指定网络上的某个特定站点；最高位为“1”其余位不为全“1”表示组播MAC地址，该地址指定网络上给定的多个站点；48位为全“1”表示广播MAC地址，该地址指定网络上所有的站点。

（4）长度/类型。

“长度/类型”（Length/Type）字段是一个二选一字段，也就是对于具体的以太帧来说它的含义不一样，占2个字节。在Ethernet I和Ethernet II以太网帧中，该字段为“类型”（Type）字段，指出帧中“数据”字段中的数据类型，这些类型的值参见表7-2，总大于1536（对应的十六进制为x600）；如果是IEEE 802.3（包括Ethernet 802.3 raw、Ethernet 802.3 SAP、802.3/802.2 LLC和802.3/802.2 SNAP）以太网帧，则该字段为“长度”（Length）字段，值总小于或等于1500（对应

的十六进制为 x5DC）。

【说明】上面的 DA、SA 和 Length/Type 这 3 个字段组成 MAC 帧头部。Pre 和 SFD 这两个字段通常不认为是 MAC 帧头部的组成部分。

（5）Data。

“数据”（Data）字段对于不同的以太网帧所包括的内容也不一样，对于 Ethernet I、Ethernet II 和 Ethernet 802.3 raw 以太网帧，它就是上层协议（可以是比以太网协议更高一点的其他数据链路层协议，也可以是网络层协议）数据；而对于 Ethernet 802.3 SAP、802.3/802.2 LLC 和 802.3/802.2 SNAP 以太网帧，则是 LLC 帧全部内容，包括 LLC 帧头和来自网络层的数据包。也正因为如此，“数据”字段长度范围也各不相同，具体如下：

- Ethernet I、Ethernet II 帧“数据”字段长度范围为 46～1500 个字节。
- Ethernet 802.3 raw 帧“数据”字段长度范围为 44～1498 个字节。
- Ethernet 802.3 SAP 和 802.3/802.2 LLC 帧“数据”字段长度范围为 43～1497 个字节。
- Ethernet 802.2 SNAP 帧“数据”字段长度范围为 38～1492 个字节。

整体而言，该字段长度范围为 38～1500 个字节。但无论怎样，总的 MAC 帧长度最小为 64 个字节，最大为 1518 个字节（不包括“前导”字段和“帧起始”字段），如果不够 64 个字节时，要在“数据”字段中加上 PAD 填充字段。

【注意】这里所说的 38～1500 个字节长度是在没有经过 IEEE 802.1q VLAN 协议重封装时的长度范围，如果封装了 VLAN 协议，则因为 VLAN 标记占用了 4 个字节，所以就整个以太网帧来说，“数据”字段的取值范围就为 34～1500 个字节。有关 IEEE 802.1q VLAN 协议将在本章后面具体介绍。

（6）Frame Check Sequence（FCS）。

“帧校验序列”（FCS）字段占 4 个字节，包括 32 位的循环冗余校验（CRC）值，由发送端对 MAC 帧中自“目的 MAC 地址”字段到“数据”字段（不包括 Pre 和 SFD 这两个字段）间的二进制序列生成校验和，然后通过接收端对所接收的帧的以上部分进行重新计算，看两次校验的结果是否一样即可得出所检验的帧在传输过程中是否已被破坏。有关 CRC 校验原理请参见第 6 章。

7.4.6 VLAN 及 QinQ 帧格式

上节介绍的 MAC 帧格式是在非 VLAN（虚拟局域网）环境下的帧格式，如果设备支持 VLAN，则在网络中传输的帧就是 VLAN 帧了。VLAN 帧会在前面介绍的非 VLAN 环境下的 MAC 帧格式中新增一些用于标识 VLAN 标签（VLAN tag）的字段。

VLAN 是由 IEEE 802.1q 标准定义的，主要用于缩小广播域，降低广播报文对网络性能的影响，同时方便对用户设备、安全策略等方面的管理，具体可参见《Cisco/H3C 交换机配置与管理完全手册》（第 3 版）。

在 VLAN 帧中都会带有一层 802.1q 标签，用于对发送该帧的设备所加入的 VLAN 进行标识，如图 7-31 所示。

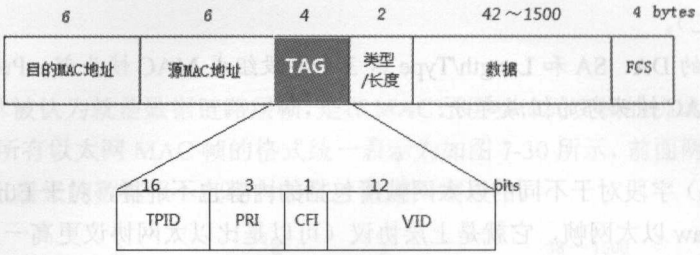


图 7-31　VLAN 帧格式

- TPID：Tag Protocol Identifier，标签协议标识，标识帧类型，占 16 位，802.1q 标签的取值为 0x8100。如果不支持 802.1q 的设备收到这样的帧，会将其丢弃。
- PRI：Priority，表示帧的二层 VLAN 优先级，占 3 位，取值范围为 0～7，值越大优先级越高。用于当交换机阻塞时优先发送优先级高的数据帧。
- CFI：Canonical Format Indicator 标准格式指示，占 1 位，0 表明 MAC 地址为标准格式（低位先传输），1 表明 MAC 地址为非标准格式（高位先传输）。在以太网中，CFI 的值为 0。
- VID：VLAN ID，占 12 位，表示该帧所属的 VLAN。

除了单层 VLAN 标签的 VLAN 帧外，还有一种带有内、外两层 VLAN 标签的 QinQ（802.1q-in-802.1q）帧，如图 7-32 所示是单层标签 VLAN 帧格式和双层标签 QinQ 帧格式的比较。

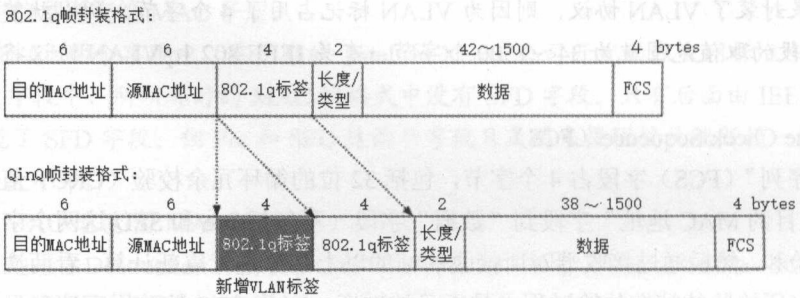


图 7-32　VLAN 帧与 QinQ 帧格式的比较

QinQ 帧带有两层 VLAN 标签的目的就是可以同时区分用户和业务类型。如一个 ISP 接入了许多企业用户，而这些企业用户中的员工业务应用又可能不一样，如有的主要进行互联网访问，有的要与外部网络进行音/视频通信，还有的只是与外部网络进行普通的资源共享访问。这时在各企业用户内部网络中为了区别不同用户类型划分了多个 VLAN，而且可能不同企业用户划分的 VLAN ID 还一样，如果直接利用企业用户自己的 VLAN，ISP 肯定承受不了，因为一则它可能没有这么多可用的 VLAN（因为可用的 VLAN 最多仅 4094 个），二则不同企业用户的 VLAN ID 分配一样就会造成冲突。

为了解决以上问题，可在 ISP 端只为每个企业用户分配一个 VLAN，并把这个 VLAN 标签添加在企业用户内部网络发出的帧中的 VLAN 标签前面（作为外层 VLAN 标签，原来帧中所带的内

部网络 VLAN 标签作为内层标签），这样就可使 ISP 设备在接收到企业用户发来的帧后可根据其外层 VLAN 标签识别是哪个企业用户发来的（因为无论何时仅能识别外层 VLAN 标签，内层 VLAN 标签当作帧的数据部分），然后为对应用户提供相应的服务。而从 ISP 网络向企业用户发送帧时又会去掉帧中原来添加的外层 VLAN 标签，恢复为只有一层 VLAN 标签的帧，这样当帧到了企业用户网络后又可根据其 VLAN 标签识别帧要发送到哪个具体的员工主机。

有关 QinQ 的具体工作原理和应用可参见《华为交换机学习指南》。

7.5 标准以太网规范及体系结构

自本节开始，就要全面、系统、深入地介绍一下我们天天打交道的以太局域网了。10Mb/s 以太网是第一个真正走入实质应用的以太网标准，尽管现在早已不再使用，但是后面的所有以太网标准都是基于这个标准改进、开发的，所以我们还是很有必要全面地了解一下这个早期的以太网标准。

7.5.1 标准以太网规范

标准以太网（也就是十兆以太网）以及后面将要介绍的快速以太网（百兆以太网）、千兆以太网和万兆以太网规范都是在 IEEE 802.3 系列标准中发布的，本节先介绍标准以太网。

标准以太网最初使用的是同轴电缆作为传输介质（此时为总线型拓扑结构），后来为了节省成本，又开发了基于双绞线标准以太网规范（此时使用集线器作为集中连接设备），再后来又为了提高传输距离而诞生了光纤标准以太网规范。

具体来说，标准以太网规范主要包括：使用 50Ω 粗同轴电缆的 10Base5（对应 IEEE 802.3 标准）、使用 50Ω 细同轴电缆的 10Base2（对应 IEEE 802.3a 标准）、使用 75Ω 同轴电缆的 10Broad36（对应 IEEE 802.3b 标准）、使用双绞线电缆的 10Base-T（对应 IEEE 802.3i 标准）和使用光纤的 10Base-F（对应 IEEE 802.3j 标准）。

这些规范名称都有其特殊的含义：对于采用同轴电缆作为传输介质的 10Mb/s 以太网规范，该委员会给出了他们所采用的规范名称格式为：

<数据传输速率（Mb/s）><信号传输模式><最大段长度（百米）>

如 10Base-5、10Base-2、10Broad36 中前面的 10 表示数据传输速率为 10Mb/s，Base 表示信号采用基带传输方式，最后面的 5、2、36 则分别表示这几种标准中单段介质的最大长度为 500m、200m（实际上是 185m）和 3.6km。但 10Base-T 有些例外，其中的 T 表示双绞线（Twisted Pairwire），10Base-F 中的 F 表示传输介质为光纤（Fiber）。

（1）10Base-5 和 10Base-2。

10Base-5（或 10Base5）和 10Base-2（或 10Base2）这两个规范都是在使用阻抗为 50Ω 同轴电缆作为传输介质的总线型拓扑结构网络中，均采用基带传输模式的曼彻斯特编码（有关这种编码的特点请参见第 4 章的相关内容）。两者的主要区别在于 10Base-5 使用粗同轴电缆（单段电缆的最大

长度为 500m），10Base-2 使用细同轴电缆（单段电缆的最大长度为 185m）。

（2）10Broad36。

10Broad36 是 IEEE 802.3 中唯一针对宽带系统（如有线广播、有线电视网络）的规范，在采用阻抗为 75Ω 同轴电缆的总线型网络拓扑结构网络中使用。它也采用基带传输模式的曼彻斯特编码，最大的端到端距离为 3600m。

（3）10Base-T。

10Base-T（或 10BaseT）规范应用于使用双绞线作为传输介质的星型拓扑结构以太网络中，是当时以太局域网中应用最广泛的以太网规范。它也采用基带传输模式的曼彻斯特编码，单段电缆的最大长度为 100m。

（4）10Base-F。

10Base-F（或 10BaseF）规范使用光纤作为传输介质，包括 10Base-FL、10Base-FB 和 10Base-FP 三种子规范，均采用基带传输模式的曼彻斯特编码，但它们的设计用途各不相同。

- 10Base-FL（或 10BaseFL）子规范设计用于与当时的 FOIRL（Fiber Optic InterRepeater Link，光纤中继器间连接）协议协同工作，其目的就是用来替换 FOIRL。在与 FOIRL 协同工作时单段光纤长度最大为 1000m，纯 10Base-FL 单段光纤长度最大为 2000m。显然，10Base-FL 是用于光纤中继器间的连接。
- 10Base-FB（或 10BaseFB）10M 光纤以太网子规范设计用于光纤骨干网（Fiber-Backbone）中同步信令的传输，用于将其他网段通过中继器连接到本地网络（不是用于用户的连接，用户光纤连接采用的是 10Base-FP 子规范），单段光纤长度最大为 2000m。显然，10Base-FB 是用于集线器或交换机与光纤中继器间的连接。
- 10Base-FP（或 10BaseFP）是一种用于连接无源光纤（Fiber-Passive）器件（这里的“无源”是指不需要外部光源）的以太网子规范，可以实现无需中继器连接大量用户（通过集线器或交换机）组成星型拓扑结构网络，单段光纤长度最大为 500m。显然，10Base-FP 是用于用户与集线器或交换机间的连接。

7.5.2 标准以太网物理层结构

因为在 IEEE 802.3 以太网中，MAC 子层是直接与物理层连接的，关系非常密切，所以把以太网中的物理层结构放在本章中介绍。

标准以太网接口的物理层和数据链路层结构（注意，这里是基于逻辑功能上的划分，不是基于物理结构上的划分）如图 7-33 所示。从图中可以看出，标准以太网的物理层包含了以下 3 个部分：MAU（介质连接单元）、AUI（连接单元接口）和 PLS（物理层信号）。

1. 介质连接单元（MAU）

MAU（Medium Attachment Unit，介质连接单元）是网络接口用来直接与传输介质（如同轴电缆、双绞线、光纤等）连接的那部分结构，发挥作用的是里面的收发器（Transceiver）芯片。在该部分中又包括了 PMA（Physical Medium Attachment，物理介质连接）和介质相关接口（Medium

Dependant Interface，MDI）两个子层的功能，在网络接口和传输介质之间提供机械连接和电气特性接口。

MDI 子层是标准以太网络接口物理层中的最低层，直接负责处理网络接口与传输介质的连接，其实就是网络接口连接器。它定义了电缆、连接电缆的连接器，以及电缆两端的终端负载的特性。因为标准以太网可以有多种不同的传输介质类型，所以也就对应有多种 MDI 连接器，这也就是 MDI 中“相关”的含义。如粗同轴电缆的 MDI 叫做插入式分接头（其实也叫 AUI 接口，如图 7-34 所示），细同轴电缆的 MDI 是 BNC 连接器（如图 7-35 所示），双绞线以太网的 MDI 叫做 RJ-45 连接器（如图 7-36 所示），光纤以太网的 MDI 可以是 ST 或 SC 连接器等（如图 7-37 和图 7-38 所示）。

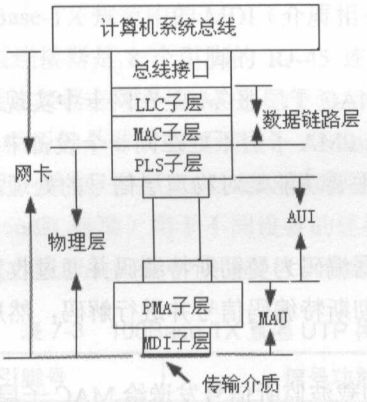


图 7-33　10Mb/s 以太网物理层和数据链路层结构

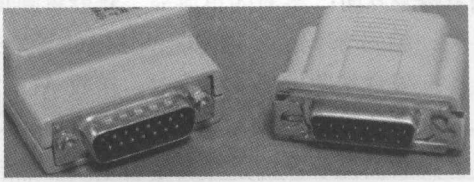


图 7-34　粗同轴电缆 AUI 连接器

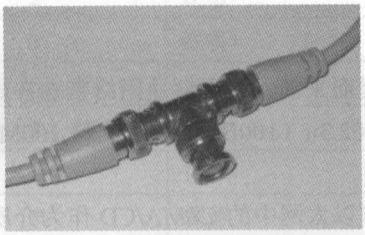


图 7-35　细同轴电缆 BNC 连接器

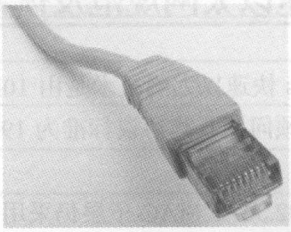


图 7-36　双绞线 RJ-45 连接器



图 7-37　ST 光纤连接器

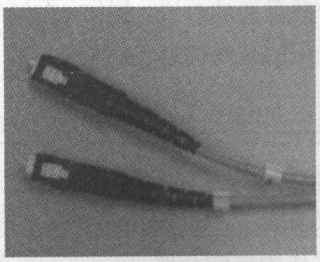


图 7-38　SC 光纤连接器

PMA 子层向 PLS 子层提供服务，负责处理与物理介质连接方面的功能（在收发器中实现），如串/并行传输的转换、检测冲突、超时控制（Jabber Control）、收发比特流。其中的“超时控制”是指当检测到某个站点发送的数据传输超过设定的最长传输时间(在芯片中有专门负责超时检测的引脚)，即认为该站点出了故障，接着就自动禁止该站向总线发送数据。

2. 连接单元接口（AUI）

AUI（Attachment Unit Interface，连接单元接口）相当于网络接口收发器上的电缆。它定义了将 MAU 与 PLS 子层相连的电缆的机械和电气特性，同时还定义了通过这个电缆所交换的信号的特性。AUI 上的信号有 4 种：发送/接收的曼彻斯特编码、冲突信号和电源信号。

【注意】如果 PLS 子层与 PMA 子层处于同一个 DTE 设备中，则不需要 AUI 和 MAU，但 DTE 与同轴电缆的 MDI 子层还是需要的。

3. 物理层信号（PLS）子层

PLS（Physical Layer Signaling，物理层信号）子层为 MAC 子层服务，是在网卡中实现的（PMA 子层和 MDI 子层是在收发器中实现的）。如果 PLS 子层与 PMA 子层不处在同一个设备中，则它通过 AUI 与 MAU 连接。从其名称可以看出，PLS 子层的主要功能是对物理层信号的处理，具体包括以下两个方面：

- 编码/解码：发送时将从 MAC 子层来的串行数据编码为曼彻斯特编码并通过收发器电缆发送给收发器；接收时，接收 AUI 发送来的曼彻斯特编码信号并进行解码，然后以串行方式发送给 MAC 子层。
- 载波监听：确定介质是否空闲，然后把监听到的载波监听信号发送给 MAC 子层。

7.6 快速以太网规范及体系结构

100Mb/s 快速以太网技术是由 10Mb/s 标准以太网发展而来的，主要解决网络带宽在局域网络应用中的瓶颈问题。其协议标准为 1995 年颁布的 IEEE 802.3u（100Base），可支持 100Mb/s 的数据传输速率。

IEEE 802.3u 在 MAC 子层仍采用了在 IEEE 802.3 标准以太网中的 CSMA/CD 作为介质访问控制协议，并保留了标准以太网的 MAC 和 LLC 帧格式。但是为了实现 100Mb/s 的传输速率，在物理层作了一些重要的改进。例如，在编码上采用了效率更高的编码方式——4B/5B 编码（在 10Mb/s 以太网中采用的是曼彻斯特编码方式，但其编码效率比较低，有效信号只占 1/2 码字）。

7.6.1 快速以太网规范

IEEE 802.3u 快速以太网标准中定义了以下 3 种不同的快速以太网规范：

- 100Base-TX：采用两对（4 根）芯线的 5 类（包括超 5 类）非屏蔽双绞线或者屏蔽双绞线作为传输介质。
- 100Base-T4：采用四对（全部的 8 根）芯线普通 3、4、5 类（包括超 5 类）双绞线作为

传输介质。

- 100Base-FX：采用光纤作为传输介质。

IEEE 802.3u 标识符包括三块信息：其一，100 表示传输速度 100Mb/s；其二，Base 表示信号仍采用“基带传输方式”；其三，是关于网络段类型的阐述：T4 表示要用四对双绞线；TX 表示要用两对双绞线；FX 表示要用两条光纤。下面是这 3 种快速以太网标准的介绍。

1. 100Base-TX

100Base-TX 快速以太网规范是采用两对芯线的双绞线电缆（可以是 5 类或超 5 类屏蔽或非屏蔽双绞线，但目前普遍采用的是成本更低更易于布线的非屏蔽双绞线）的，其中一对用于发送数据，另一对用于接收数据。该标准直接用于取代标准以太网中的 10Base-T 和 10Base-2 规范。

100Base-TX 规范中的 MDI（介质相关接口）连接器有两种：对于非屏蔽双绞线（UTP）电缆，MDI 子层连接器是 8 个引脚的 RJ-45 连接器（俗称“水晶头”）；对于屏蔽双绞线（STP）电缆，MDI 子层连接器必须是 IBM 的 STP 连接器。

如果使用的是非屏蔽双绞线，100Base-TX 规范中只使用了 4 对芯线中的 2 对，另外 2 对芯线没有使用。根据 EIA/TIA-568B 布线标准，直通双绞网线（两端同时用 TIA/EIA-568A 或者 TIA/EIA-568B 标准）用于不同设备的连接，如 PC 机到集线器、交换机的连接，交换机到路由器的连接等，其水晶头的 8 个引脚与双绞线 8 条芯线的连接如表 7-3 所示。

表 7-3 100Base-TX 直通 UTP 网线水晶头引脚、信号功能与双绞芯线的连接分配表

引脚号	信号功能	芯线颜色
1	发送+	橙白色
2	发送-	橙色
3	接收+	绿白色
4	保留	—
5	保留	—
6	接收-	绿色
7	保留	—
8	保留	—

100Base-TX 标准也支持特征阻抗为 150Ω 的 5 类屏蔽双绞线。屏蔽双绞线电缆使用 D 型连接器。直通 STP 网线 DB-9 连接器上管脚与双绞芯线的连接如表 7-4 所示。

表 7-4 100Base-TX 的 STP MDI 连接器引脚与双绞芯线的连接分配表

引脚号	信号功能	芯线颜色
10	公共地	电缆外壳
1	接收+	橙色
2	保留	—

续表

引脚号	信号功能	芯线颜色
3	保留	—
4	保留	—
5	发送+	红色
6	接收-	黑色
7	保留	—
8	保留	—
9	发送-	绿色

当要使用交叉电缆时（如同类设备普通端口上的级联，如 PC 之间、交换机之间、路由器之间等），按照表 7-5 所示的跳线规则进行交叉网线（一端用 TIA/EIA-568A 标准，另一端用 TIA/EIA-568B 标准，用于同种设备的连接，如 PC 到 PC 的连接、交换机普通端口间的级连、PC 到路由器的连接、路由器与路由器间的连接等）制作。

目前的以太网端口基本上都支持电缆跳线自动翻转功能（即 MDI/MDIX 自动识别功能），在任何情况下都既可以使用直通网线，又可以使用交叉网线，所以目前一般都是只制作直通网线即可。有关 MDI 和 MDIX 接口参见第 5 章中的介绍。

表 7-5　100Base-TX 交叉网线引脚分配对比表

引脚号	5 类 UTP 电缆芯线颜色		5 类 STP 电缆芯线颜色	
	一端颜色	另一端颜色	一端颜色	另一端颜色
1	橙白色	绿白色	橙色	红色
2	橙色	绿色	—	—
3	绿白色	橙白色	—	—
4	—	—	—	—
5	—	—	红色	橙色
6	绿色	橙色	黑色	绿色
7	—	—	—	—
8	—	—	—	—
9	N/A	N/A	绿色	黑色
10	N/A	N/A	公共地	公共地

从表 7-5 中可以看出，在使用非屏蔽双绞线（UTP）制作交叉网线时两端的芯线呈以下关系：

- 一端的 1 号引脚所对应的芯线与另一端的 3 号引脚所对应的芯线一样，即 1 号引脚与 3 号引脚对调。

- 一端的 2 号引脚所对应的芯线与另一端的 6 号引脚所对应的芯线一样，即 2 号引脚与 6 号引脚对调。

在使用屏蔽双绞线（STP）制作交叉网线时两端的芯线呈以下关系：

- 一端的 1 号引脚所对应的芯线与另一端的 5 号引脚所对应的芯线一样，即 1 号引脚与 5 号引脚对调。
- 一端的 6 号引脚所对应的芯线与另一端的 9 号引脚所对应的芯线一样，即 6 号引脚与 9 号引脚对调。

2. 100Base-T4

100Base-T4 是用来向下兼容那些已经安装了 3 类或 4 类非屏蔽双绞线电缆（当然也可以使用 5 类或超 5 类双绞线）的快速以太网规范，但它要求同时使用双绞线电缆中的全部 4 对（8 根）芯线，这也就是“T4”的由来。

100Base-T4 规范采用半双工交换方式，在它的 4 对芯线中的 3 对用于一起发送数据，第 4 对用于冲突检测（其目的就是为了在 3 类或 4 类非屏蔽双绞线电缆中实现 100Mb/s 的传输速率）。其水晶头引脚与双绞芯线的连接分配如表 7-6 所示。

表 7-6 100Base-T4 水晶头引脚、信号功能与双绞芯线的连接分配表

引脚号	信号功能	芯线颜色
1	TX_D1+	橙白色
2	TX_D1-	橙色
3	RX_D2+	绿白色
4	BI_D3+	蓝色
5	BI_D3-	蓝白色
6	RX_D2-	绿色
7	BI_D4+	棕白色
8	BI_D4-	棕色

表 7-7 所示是 100Base-T4 规范中的交叉双绞网线水晶头引脚与双绞芯线的连接分配表。

表 7-7 100Base-T4 交叉双绞网线水晶头引脚与双绞芯线的连接分配表

引脚号	一端芯线颜色	另一端芯线颜色
1	橙白色	绿白色
2	橙色	绿色
3	绿白色	橙白色
4	蓝色	棕白色
5	蓝白色	棕色

续表

引脚号	一端芯线颜色	另一端芯线颜色
6	绿色	橙色
7	棕白色	蓝色
8	棕色	蓝白色

从表 7-7 中可以看出，在制作 100Base-T4 规范交叉双绞线时两端的芯线呈以下关系：

- 一端的 1 号引脚所对应的芯线与另一端的 3 号引脚所对应的芯线一样，即 1 号引脚与 3 号引脚对调。
- 一端的 2 号引脚所对应的芯线与另一端的 6 号引脚所对应的芯线一样，即 2 号引脚与 6 号引脚对调。
- 一端的 4 号引脚所对应的芯线与另一端的 7 号引脚所对应的芯线一样，即 4 号引脚与 7 号引脚对调。
- 一端的 5 号引脚所对应的芯线与另一端的 8 号引脚所对应的芯线一样，即 5 号引脚与 8 号引脚对调。

在 100Base-T4 规范中，原来的(1,3)和(2,6)这两对双绞芯线与 100Base-TX 规范一样仍只能半双工传输，但另外两对双绞芯线可以全双工传输，如图 7-39 所示。

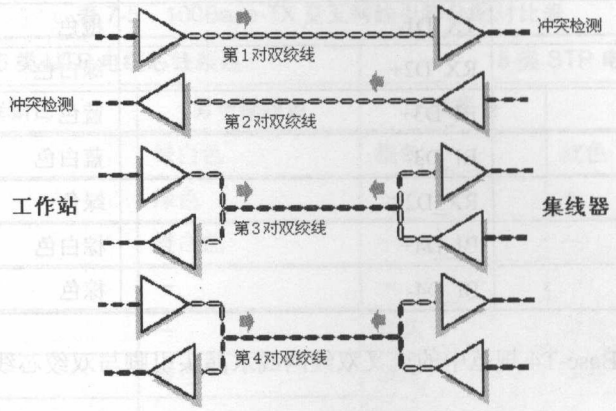


图 7-39　100Base-T4 规范中的双绞芯线使用情况

当工作站传送数据给集线器或交换机时，第 1、3、4 对双绞线用来传送数据，第 2 对双绞线用来检测冲突；当集线器或交换机传送数据给工作站时，第 2、3、4 对双绞线用来传送数据，第 1 对双绞线用来检测冲突。也就是说，每一个方向的传送都同时使用了 3 对双绞线来传送数据。这样一来，对于 100Mb/s 的传送速率来说，每一对双绞线的传送速率只要 33.33Mb/s，更加容易实现。另外，100Base-T4 规范所采用的是 8B/6T 编码方法（不再是其他快速以太网规范所使用的 4B/5B 编码方式），至于这些编码方式这里不作介绍，大家可以查看其他相关资料。

3. 100Base-FX

100Base-FX 快速以太网规范采用的是两条光纤（单工模式），一条用于发送数据，一条用于接收数据（现在新型的光纤连接方式是仅需一条光纤，通过波分复用技术可实现在同一条光纤上同时发送和接收数据）。当工作站的光纤网卡以全双工模式运行时能超过 2km。100Base-FX 规范中可使用的光纤有以下两类（有关光纤分类的详细知识请参见第 5 章）：

- 多模光纤：这种光纤的纤径为 62.5/125μm，采用基于 LED（发光二极管）的收发器将波长为 820nm 的光信号发送到光纤上。当连在两个设置为全双工模式的交换机端口之间时，支持的最大距离为 2km。
- 单模光纤：这种光纤的纤径为 9/125μm，采用基于激光的收发器将波长为 1300nm 的光信号发送到光纤上。单模光纤损耗小，较之多模光纤能使光信号传输到更远的距离。

7.6.2 快速以太网物理层结构

快速以太网与前面介绍的标准以太网在物理层结构上存在比较大的区别，具体对比如图 7-40 所示。下面仍以由低到高的顺序介绍这些逻辑组成部分。

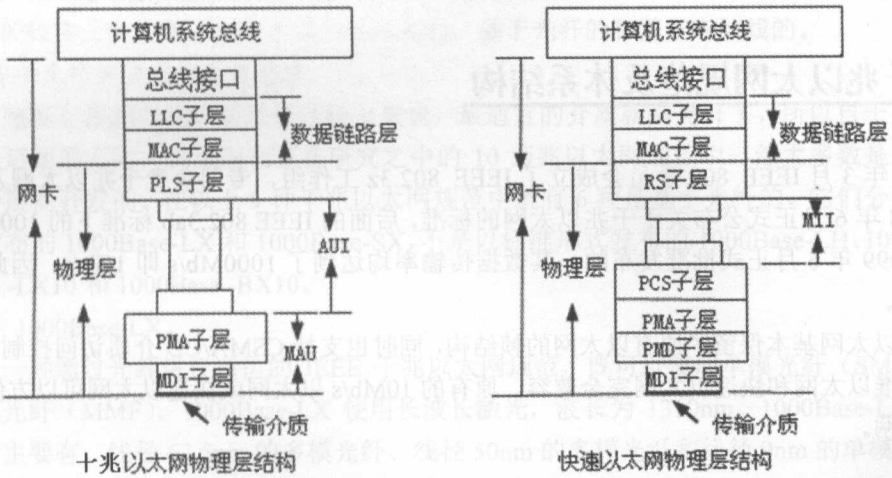


图 7-40 标准以太网与快速以太网的体系结构比较

（1）介质相关接口（Medium Dependent Interface，MDI）子层。

在标准以太网物理层结构中最低层也是 MDI 子层，而且功能也是一样的，规定 PMD 子层和传输介质之间的连接器类型，如 100Base-TX 的 RJ-45 连接器、100Base-FX 的 SC 和 ST 连接器等。

（2）物理介质相关（Physical Media Dependent，PMD）子层。

PMD 子层是快速以太网物理层结构中新增加的一层，位于收发器上。它与物理介质直接相连的是信号收发器和信号检测模块，主要提供信号的收发、检测和编/解码。发送时要将来自 PMA 子层的信号经过适当的编码转换成适合特定传输介质的信号，并提供发送信号驱动；接收信号时，

PMD 子层经过相应的解码处理后发送给 PMA 子层。

（3）物理介质连接（Physical Medium Attachment，PMA）子层。

PMA 子层完成链路监测、载波检测、NRZI（非归零翻转）编/解码和发送时钟合成、接收时钟恢复的功能。

（4）物理编码子层（Physical Coding Sublayer，PCS）。

PCS 子层的主要功能是 4B/5B 编/解码（而不是标准以太网中的曼彻斯特编码，当然在 100Base-T4 规范中使用的是 8B/6T 编/解码方式）、碰撞检测和并串转换。

（5）MII。

MII（Medium Independent Interface，介质无关接口）逻辑上与 10Mbps 以太网的 AUI 接口对应，可使 MAC 子层与传输介质无关。MII 在发送和接收数据时，由原来标准以太网 AUI 的一位位的串行传输改变为半字节（4 位）的并行传输，这样发送和接收时钟只需整个数据传输速率的 1/4，即 25MHz，更容易实现，也更稳定。

（6）协调子层（Reconciliation Sublayer，RS）。

RS 子层是替换原来标准以太网的 PLS 子层。PLS 子层的功能是对物理层信号进行编/解码和载波检测，而快速以太网的 RS 子层是将 MAC 子层的业务定义映射成 MII 的信号。

7.7 千兆以太网规范及体系结构

1996 年 3 月 IEEE 802 委员会成立了 IEEE 802.3z 工作组，专门负责千兆以太网及其标准，并于 1998 年 6 月正式公布关于千兆以太网的标准，后面的 IEEE 802.3ab 标准下的 1000Base-T 规范是在 1999 年 6 月正式批准发布的，其数据传输率均达到了 1000Mb/s 即 1Gb/s，因此也称吉比特以太网。

千兆以太网基本保留了原有以太网的帧结构，同时也支持 CSMA/CD 介质访问控制技术，所以向下与标准以太网和快速以太网完全兼容，原有的 10Mb/s 以太网或快速以太网可以方便地升级到千兆以太网。

7.7.1 千兆以太网规范

最早在 1998 年和 1999 年发布的 IEEE 802.3z 和 IEEE 802.3ab 标准中就包括了：1000Base-LX、1000Base-SX、1000Base-CX 和 1000Base-T（前 3 种统称为 1000Base-X 子系列），如图 7-41 所示。其中前 3 个是由 IEEE 802.3z 标准规定的，而 1000 Base-T 规范是由 IEEE 802.3ab 标准规定的，是后面专门开发的。这 4 个千兆以太网规范支持不同类型的光纤和双绞线电缆。

除了以上 4 种以标准形式发布的 IEEE 千兆以太网规范外，在工业应用中还有一些并没有正式以标准形式对外发布但却实实在在有广泛应用的千兆以太网规范，如 1000Base-LH、1000Base-ZX、1000Base-LX10、1000Base-BX10、1000Base-TX 这 5 种规范。

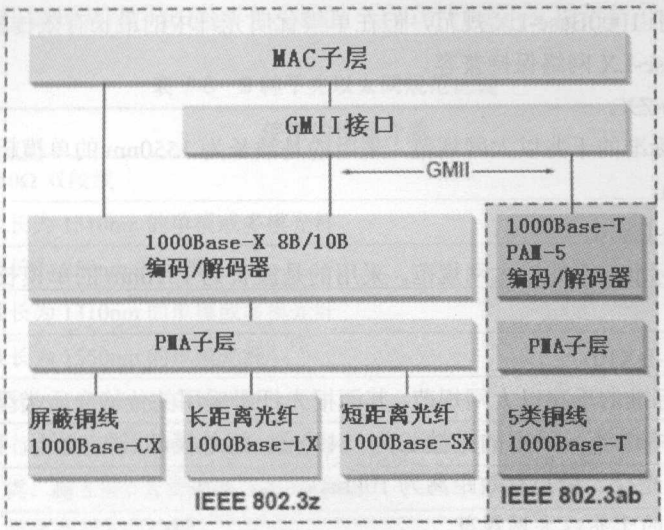


图 7-41 千兆以太网体系结构

这样一来，在千兆以太网系列中加起来一共就有 9 种规范了。在这 9 种千兆以太网规范中，根据所采用的传输介质类型，总体上是分两大类的：基于光纤的和基于双绞线的。

1. 基于光纤的千兆以太网规范

千兆速率已经相当高，从总体性能上来说，最适宜的介质就是光纤了，所以自千兆以太网以后，包括后面的万兆，甚至现在正在研究之中的 10 万兆以太网规范中，绝大多数是基于光纤这种传输介质而开发的。在以上 9 种千兆以太网规范中就有 6 种是基于光纤的，它们分别是已以标准形式发布的 1000Base-LX 和 1000Base-SX，不是以标准形式发布的 1000Base-LH、1000Base-ZX、1000Base -LX10 和 1000Base -BX10。

（1）1000Base-LX。

这是一种通过光纤进行通信的 IEEE 千兆以太网规范，既可以使用单模光纤（SMF），也可以使用多模光纤（MMF）。1000Base-LX 使用长波长激光，波长为 1310nm。1000Base-LX 规范所使用的光纤主要有：线径 62.5nm 的多模光纤、线径 50nm 的多模光纤和线径 9nm 的单模光纤。其中使用多模光纤的最大传输距离为 550m，使用单模光纤的最大传输距离为 5km。1000Base-LX 规范采用 8B/10B 数据编码方式，主要适用于校园网或城域网的主干网。

（2）1000Base-SX。

这也是一种通过光纤进行通信的 IEEE 千兆以太网规范，适用于线径为 50nm 和 62.5nm 的短波（波长为 850nm）多模光纤（MMF）。其中使用 62.5nm 多模光纤的最大传输距离为 275m，使用 50nm 多模光纤的最大传输距离为 550m。1000Base-SX 规范也采用 8B/10B 数据编码方式，适合作为大楼网络系统的主干通路。

（3）1000Base-LH。

这是一个非标准的千兆以太网规范，采用的是波长为 1300nm，或者 1310nm 的单模或者多模

长波光纤。它类似于 1000Base-LX 规范，但在单模优质光纤中的最长有效传输距离可达 10 公里，并且可以与 1000Base-LX 网络保持兼容。

（4）1000Base-ZX。

这也是一个非标准的千兆以太网规范，采用的是波长为 1550nm 的单模超长波光纤，最长有效传输距离可达 70km。

（5）1000Base-LX10。

这也是一个非标准的千兆以太网规范，采用的是波长为 1310nm 的单模长波光纤，最长有效传输距离可达 10km。

（6）1000Base-BX10。

这也是一个非标准的千兆以太网规范，其两根光纤所采用的传输介质类型是不同的：下行方向（从网络中心到网络边缘）采用的是波长为 1490nm 的单模超长波光纤，而上行方向则是采用 1310nm 的单模长波光纤，最长有效距离为 10km。

2. 基于双绞线的千兆以太网规范

采用双绞线作为传输介质的千兆以太网规范有以下 3 个：

（1）1000Base-CX。

这是一种采用 150Ω 平衡屏蔽双绞线（STP）作为传输介质（连接器为 DB-9）的千兆以太网规范，传输距离最长仅为 25m，数据编码方法采用 8B/10B，适用于数据中心设备间的连接（如交换机之间的连接，尤其适用于主干交换机和主服务器之间的短距离连接）或者堆叠设备间的短距离互连，但不适用于数据中心与配线架的连接。

（2）1000Base-T。

这是一种可以采用 5 类、超 5 类、6 类或者 7 类 UTP 双绞线的全部 4 对芯线作为传输介质的千兆以太网规范，对应标准为 IEEE 802.3ab（与其他千兆以太网标准不同）。它的最大传输距离为 100m。在全部的 4 对双绞芯线中，每对都可以同时进行全双工数据收发，所以即使是相同设备间的连接也无需制作交叉线，两端都用相同的布线标准即可。这是目前在企业局域网中最常用的一种千兆以太网标准。

（3）1000Base-TX。

在千兆以太网标准中还有一种常用的标准，即 1000Base-TX，但它不是由 IEEE 制作的，而是由 TIA/EIA 于 1995 年发布的，对应的标准号为 TIA/EIA-854。

尽管 1000Base-TX 也是基于 4 对双绞线，但却采用快速以太网中与 100Base-TX 标准类似的传输机制，以两对线发送，两对线接收（类似于 100Base-TX 的一对线发送，一对线接收）。由于每对线缆本身不同时进行双向的传输，所以线缆之间的串扰大大降低，同时其编码方式也是 8B/10B。这种技术对网络的接口要求比较低，不需要非常复杂的电路设计，降低了网络接口的成本。但由于使用线缆的效率降低了（两对线收，两对线发），要达到 1000Mb/s 的传输速率，就要求带宽超过 100MHz，也就是说在 5 类和超 5 类的系统中不能支持这种类型的网络，一定需要 6 类或者 7 类双绞线系统的支持。

以上 9 种千兆以太网规范的比较如表 7-8 所示，从中可以看出各规范的主要优势和特性。

表 7-8 9 种千兆以太网规范比较

千兆以太网规范	使用的传输介质	有效距离
1000Base-CX	150Ω 双绞线	25m
1000Base-LX	波长为 1310nm 的单模或多模光纤	5km
1000Base-SX	波长为 850nm 的多模光纤	500m
1000Base-LH	波长为 1310nm 的单模或多模光纤	10km
1000Base-ZX	波长为 1550nm 的单模光纤	70km
1000Base-LX10	波长为 1310nm 的单模光纤	10km
1000Base-BX10	下行为波长为 1490nm 的单模光纤，上行为波长为 1310nm 的单模光纤	10km
1000Base-T	5 类、超 5 类、6 类或者 7 类双绞线	100m
1000Base-TX	6 类或者 7 类双绞线	100m

7.7.2 1000Base-T 以太网技术

在以上 9 种千兆以太网规范中，性价比最高的就是 1000Base-T 这种采用普通 5 类以上双绞线的千兆以太网规范。尽管在双绞线千兆以太网规范中还有一种 1000Base-TX 规范，但是它只能使用 6 类以上的双绞线，网络建设成本明显要高于 1000Base-T 规范。所以在此专门介绍一下这种应用最广的 1000Base-T 千兆以太网规范。

1000Base-T 是专门为在 5 类双绞线上进行千兆速率数据传输而设计的。它采用了双绞线的全部 4 对芯线，并且是全双工传输的，也就是每对双绞线都可以同时进行数据的发送和接收，这样一来 1Gb/s 的传输速率可以等效地看作在 4 对双绞线上，每对的传送速率为 250Mb/s（1000Mb/s÷4 = 250Mb/s）。

因为 1000Base-T 只支持全双工传输，所以与 1000Base-T 千兆以太网端口直接相连的端口也必须是支持全双工的以太网端口（最佳情况则是同时为 1000Base-T 千兆以太网端口），而不能是半双工的，否则一方面性能会严重下降，达不到千兆的效果，另一方面还会有严重的丢包现象。相比之下，另一个千兆以太网规范——1000Base-TX 尽管也是采用了全部的 4 对双绞芯线，但是它同一时间是两对（4 根）芯线用于发送，另两对（4 根）芯线用于接收，属于半双工模式，类似于快速以太网中的 100Base-TX 规范，只不过 100Base-TX 规范中是同一时间发送数据和接收数据都仅用一对（2 根）芯线。

1000Base-TX 和 1000Base-T 规范双绞芯线的使用情况比较如图 7-42 所示（注意区分线中的不同颜色）。1000Base-T 规范中各芯线的具体作用如图 7-43 所示，每条芯线上分担的速率都是 250Mb/s，所以在全双工模式下它可以实现 2Gb/s 的传输速率。

1000Base-T 采用了一种更加强大的信号传输和编/解码方案——PAM-5（Pulse Amplitude

Modulation 5，脉冲调幅-5）。PAM-5 编码使用-2、-1、0、+1、+2 五种电平，其中-2、-1、+1、+2 这 4 个电平用于信号编码，0 电平用于前向纠错编码（FEC）。而在 100Base-TX 中采用的是 3 级 MLT（3Multi-Level Transmission，多级传送）编码方案 MLT-3，使用-1、0、+1 三种电平，其中-1、+1 这两个电平用于信号编码，0 电平用于前向纠错编码。由此可见，PAM-5 编码方案可以在链路上较 MLT-3 编码方案多传送一倍的数据。

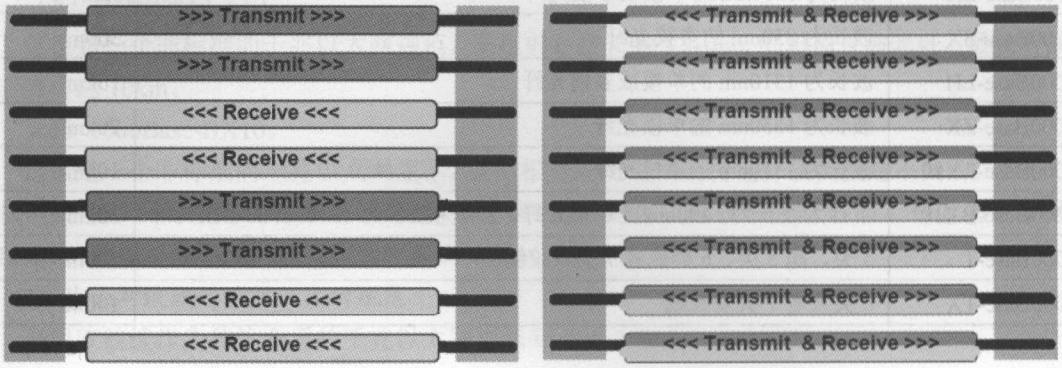


图 7-42　100Base-TX 和 100Base-T 规范中双绞芯线使用比较

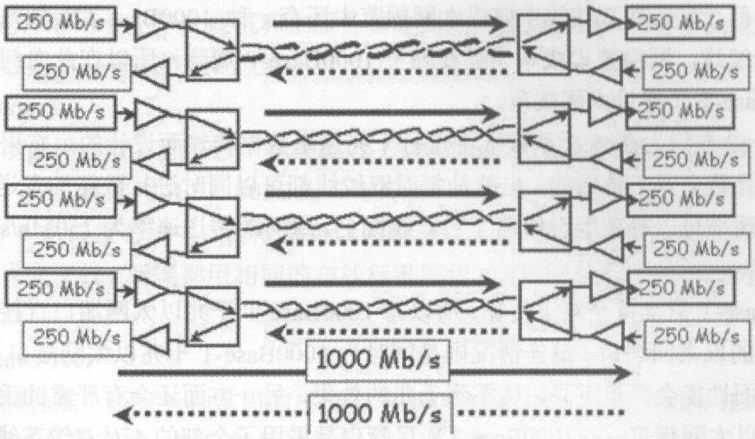


图 7-43　100Base-T 规范中各双绞芯线的作用

另外，1000Base-T 规范中不采用交叉网线，只需直通线（两端按同一标准制作网线）即可，因为这一规范中总是采取全双工传输，任何一根网线都可以同时发送和接收数据。这与 1000Base-TX 是不一样的（有“X”的表示不同设备连接时要采用交叉线，而没有“X”的表示不用交叉线）。

总体而言，1000Base-T 规范最吸引人的地方在于为企业提供了一种除多模光纤以太网方案外更廉价的千兆方案，用户可以在原来 100Base-T 的基础上平滑升级到 1000Base-T。该规范主

要用于结构化布线中同一层建筑的通信，可以利用现有以太网或快速以太网已铺设的 UTP 电缆进行网络升级，也可被用作大楼内的网络主干，大大节省了成本。这是目前最主要应用的千兆以太局域网方案。

7.7.3 IEEE 千兆以太网物理层结构

尽管在以上9种千兆以太网中有些并不是IEEE发布的，但总体上来说，它们的体系结构与IEEE发布的千兆以太网规范类似，所以在此仅以 IEEE 发布的千兆以太网标准为例介绍其体系结构。

与本章前面介绍的标准以太网、快速以太网物理层结构一样，千兆以太网的整个体系结构的差别主要体现在物理层和 MAC 子层，LLC 子层基本上都保持了一致。从体系结构上来看，千兆以太网与快速以太网的区别并不是很大，主要就是由原来的 MII 替换成了 GMII。如图 7-44 所示是快速以太网体系结构与千兆以太网体系结构的比较。

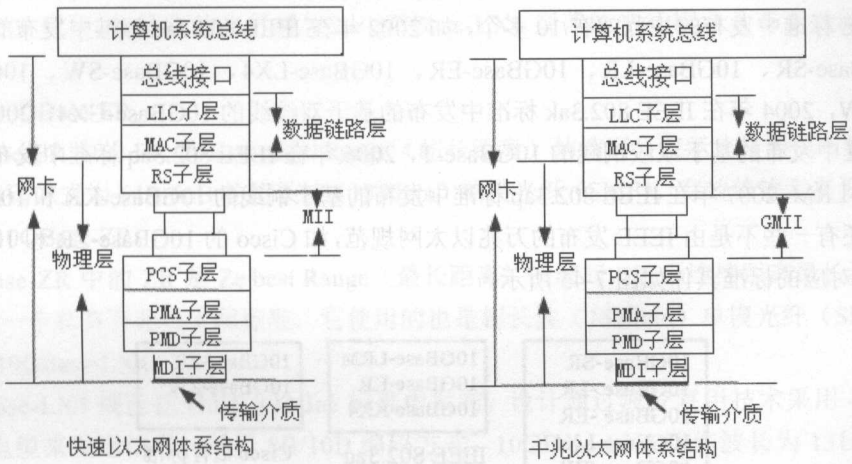


图 7-44 快速以太网与千兆以太网体系结构的比较

快速以太网体系结构与千兆以太网体系结构在物理层上的主要区别表现在RS子层和PCS子层之间的接口发生了改变：由快速以太网中的介质无关接口（MII）扩展为千兆介质无关接口（GMII）。其他各子层的功能参见本章前面介绍的快速以太网标准中的物理层结构。

GMII 的发送和接收数据宽度由原来的 MII 的半字节（4 位）扩展到一个字节（8 位），这样每根芯线使用 125MHz 的时钟频率可以实现 1000Mb/s 的传输速率。另外与 MII、AUI 不同，GMII 不支持连接器和电缆，只是内置作为 IC 和 IC 之间的接口。

另外，在千兆以太网物理层的 PCS 子层中，却与快速以太网中的 PCS 子层有所区别，而且 1000Base-X 子系列 3 个规范物理层与 1000Base-T 规范物理层中的 PCS 子层都不一样，前者采用的是 8B/10B 编码方式，后者采用的是 PAM-5 编码方式，而快速以太网 100Base-TX 规范中采用的是 4B/5B 编码方式。

7.8 万兆以太网规范及体系结构

1999 年底成立了 IEEE 802.3ae 工作组进行万兆以太网（Ten Gigabit Ethernet）技术（10Gb/s）的研究，并于 2002 年以 IEEE 802.3ae 标准的形式第一次发布万兆以太网标准，是当时最快的以太网标准。在 10G 以太网中，最显著的改变就是只能与全双工交换机连接，不再支持半双工连接，也不支持 CSMA/CD。

7.8.1 万兆以太网规范

万兆以太网标准和规范都比较繁多：在标准方面，有 2002 年的 IEEE 802.3ae、2004 年的 IEEE 802.3ak、2006 年的 IEEE 802.3an 和 IEEE 802.3aq，以及 2007 年的 IEEE 802.3ap；在规范方面，仅由上述 IEEE 标准中发布的规范就有 10 多个，如 2002 年在 IEEE 802.3ae 标准中发布的基于光纤的规范 10GBase-SR、10GBase-LR、10GBase-ER、10GBase-LX4、10GBase-SW、10GBase-LW、10GBase-EW，2004 年在 IEEE 802.3ak 标准中发布的基于双绞线的 10GBase-CX4，2006 年在 IEEE 802.3an 标准中发布的基于双绞铜线的 10GBase-T，2006 年在 IEEE 802.3aq 标准中发布的基于光纤的 10GBase-LRM，2007 年在 IEEE 802.3ap 标准中发布的基于铜线的 10GBase-KR 和 10GBase-KX4。除此之外，还有一些不是由 IEEE 发布的万兆以太网规范，如 Cisco 的 10GBase-ZR 和 10GBase-ZW，这些规范所对应的标准具体如图 7-45 所示。

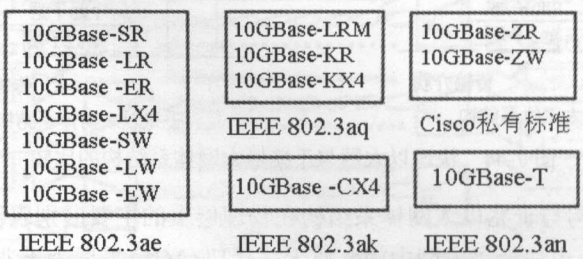


图 7-45 万兆以太网标准与对应的规范

以上这 10 多种万兆以太网规范可以分为 3 类：一是基于光纤的局域万兆以太网规范；二是基于双绞线（或铜线）的局域万兆以太网规范；三是基于光纤的广域万兆以太网规范。

1. 基于光纤的局域万兆以太网规范

用于局域网的光纤万兆以太网规范有：10GBase-SR、10GBase-LR、10GBase-LRM、10GBase-ER、10GBase-ZR 和 10GBase-LX4 这 6 个。

（1）10GBase-SR。

10GBase-SR 中的 SR 是 Short Range（短距离）的缩写，表示仅用于短距离连接。该规范支持

编码方式为64B/66B的短波（波长为850nm）多模光纤（MMF），有效传输距离为2m～300m。但要支持300m传输需要采用经过优化的50 μm线径OM3（Optimized Multimode 3，优化的多模3）光纤（没有优化的线径50 μm光纤称为OM2光纤，而线径为62.5μm的光纤称为OM1光纤）。

（2）10GBase-LR。

10GBase-LR中的LR是Long Range（长距离）的缩写，表示主要用于长距离连接。该规范支持编码方式为64B/66B的长波（1310nm）单模光纤（SMF），有效传输距离为2m～10km，事实上最高可以达到25km。

（3）10GBase-LRM。

10GBase-LRM中的LRM是Long Reach Multimode（长距离延伸多点模式）的缩写，表示主要用于长距离的多点连接模式，对应的标准为2006年发布的IEEE 802.3aq，采用64B/66B编码方式。采用该规范时，在1990年以前安装的FDDI 62.5μm多模光纤FDDI网络和100Base-FX网络中的有效传输距离为220m，而在OM3光纤中可达260m，在连接长度方面不如以前的10GBase-LX4规范，但是它的光纤模块比10GBase-LX4规范的光纤模块具有更低的成本和电源消耗。

（4）10GBase-ER。

10GBase-ER中的ER是Extended Range（超长距离）的缩写，表示连接距离可以非常长。该规范支持编码方式为64B/66B的超长波（1550nm）单模光纤（SMF），有效传输距离为2m～40km。

（5）10GBase-ZR。

10GBase-ZR中的ZR是Ze best Range（最长距离）的缩写，表示连接距离最长，可达80km，是Cisco的一个私有万兆以太网标准。它使用的也是超长波（1550nm）单模光纤（SMF）。

（6）10GBase-LX4。

10GBase-LX4规范在IEEE 802.3ae标准中发布，设计通过波分复用技术采用4束光波通过单对光学电缆来发送信号，采用8B/10B编码方式。10GBase-LX4工作波长为1310nm，使用多模或单模暗光纤，主要适用于需要在一个光纤模块中同时支持多模和单模光纤的环境。多模光纤传输距离为240m～300m，单模光纤传输距离10km以上，根据电缆的类型和质量，还能达到更远的距离。

2. 基于双绞线（或铜线）的局域万兆以太网规范

在2002年发布的几个万兆以太网规范中并没有支持铜线这种廉价传输介质的，但事实上，像双绞线这类铜线在局域网中的应用是最普遍的，不仅成本低，而且容易维护，所以在近几年就相继推出了多个基于双绞线（6类以上）的万兆以太网规范，即：10GBase-CX4、10GBase-KX4、10GBase-KR和10GBase-T。

（1）10GBase-CX4。

10GBase-CX4规范使用IEEE 802.3ae中定义的XAUI（万兆附加单元接口）和用于InfiniBand中的4X连接器，传输介质称为“CX4铜缆”（其实就是一种屏蔽双绞线）。它的有效传输距离仅为15m。

10GBase-CX4 规范不是利用单个铜线链路传送万兆数据，而是使用 4 台发送器和 4 台接收器来传送万兆数据，并以差分方式运行在同轴电缆上，每台设备利用 8B/10B 编码以每信道 3.125GHz 的波特率传送 2.5Gb/s 的数据。这需要在每条电缆组的总共 8 条双同轴信道的每个方向上有 4 组差分线缆对。另外，与可在现场端接的 5 类、超 5 类双绞线不同，CX4 线缆需要在工厂端接，因此客户必须指定线缆长度。线缆越长一般直径就越大。

（2）10GBase-KX4 和 10GBase-KR。

10GBase-KX4 和 10GBase-KR 两个规范主要用于设备背板连接中，如刀片服务器、路由器和交换机的集群线路卡，所以又称之为“背板以太网”。

万兆背板连接目前已经存在并行和串行两种版本：并行版本（10Gbase-KX4 规范）是背板的通用设计，将万兆信号拆分为 4 条通道（类似 XAUI），每条通道的带宽都是 3.125Gb/s；串行版本（10GBase-KR 规范）中只定义了一条通道，采用 64/66B 编码方式实现 10Gb/s 高速传输。10Gbase-KX4 使用与 10GBase-CX4 规范一样的物理层 8B/10B 编码，10GBase-KR 使用与 10GBase-LR/ER/SR 这 3 个规范一样的物理层 64B/66B 编码。

（3）10GBase-T。

10GBase-T 是基于屏蔽或非屏蔽双绞线，主要用于局域网的万兆以太网规范，最长传输距离为 100m。这可以算是万兆以太网的一项革命性进步，因为在此之前，一直认为在双绞线上不可能实现这么高的传输速率，原因就是运行在这么高工作频率（至少为 500MHz）基础上的损耗太大。但标准制定者依靠以下 4 项技术构件使 10GBase-T 变为了现实：损耗消除、模拟到数字转换、线缆增强和编码改进。

在编码方面，10GBase-T 不是采用原来 1000Base-T 的 PAM-5，而是采用 PAM-8 编码方式，支持 833Mb/s 和 400MHz 带宽，对布线系统的带宽要求也相应地修改为 500MHz，而如果仍采用 PAM-5 的 10GBase-T 则对布线带宽的需求是 625MHz；在连接器方面，10GBase-T 使用已广泛应用于以太网的 650 MHz 版本 RJ-45 连接器。在 6 类线上最长有效传输距离为 55m，而在 6a 类双线上可以达到 100m。

3. 基于光纤的广域万兆以太网规范

10G 以太网一个最大的改变就是它不仅可以在局域网中使用，还可以应用于广域网中，其对应的规范包括：10GBase-SW、10GBase-LW、10GBase-EW 和 10GBase-ZW（此为 Cisco 私有标准）。这 4 个 10G 广域以太网规范专为工作在 OC-192/STM-64 SDH/SONET 环境而设置，使用 SDH（Synchronous Digital Hierarchy，同步数字体系）/SONET（Synchronous Optical Networking，同步光纤网络）帧，传输速率为 9.953Gb/s。它们所使用的光纤类型和有效传输距离分别对应于前面介绍的应用于局域网中的 10GBase-SR、10GBase-LR、10GBase-ER 和 10GBase-ZR 规范。

表 7-9 综合了以上介绍的所有 10G 以太网规范，在实际的网络系统设计中就可以针对具体的节点环境和网络需求来对应选择了。当然，目前 10G 以太网主要是在骨干网中采用，在一般的企业局域网中仍很少见到。

表 7-9 万兆以太网规范比较

万兆以太网规范	使用的传输介质	有效距离	应用领域
10GBase-SR	850nm 多模光纤、50μm 的 OM3 光纤	300m	局域网
10GBase-LR	1310nm 单模光纤	10km	
10GBase-LRM	62.5 μm 多模光纤、OM3 光纤	260m	
10GBase-ER	1550nm 单模光纤	40km	
10GBase-ZR	1550nm 单模光纤	80km	
10GBase-LX4	1300nm 单模或多模光纤	300m（多模时）、10km（单模时）	
10GBase-CX4	屏蔽双绞线	15m	
10GBase-T	6 类、6a 类双绞线	55m（6 类线时）、100m（6a 类线时）	
10GBase-KX4	铜线（并行接口）	1m	背板以太网
10GBase-KR	铜线（串行接口）	1m	
10GBase-SW	850nm 多模光纤、50μm 的 OM3 光纤	300m	SDH/SONET 广域网
10GBase-LW	1310nm 单模光纤	10km	
10GBase-EW	1550nm 单模光纤	40km	
10GBase-ZW	1550nm 单模光纤	80km	

7.8.2 万兆以太网物理层结构

在许多万兆以太网规范中也对应了许多不同类型的万兆以太网物理层，但总体类型还是与最初于 2002 年发布的几类万兆以太网规范差不多。下面分基于光纤传输介质万兆以太网规范物理层和基于铜线传输介质万兆以太网规范物理层两种类型进行介绍。

在 IEEE 发布的万兆以太网规范中，主要是三大类，即：10GBase-X（仅包括 10GBase-X 规范）、10GBase-R（包括 10GBase-SR、10GBase-LR 和 10GBase-ER 三个规范）和 10GBase-W（包括 10GBase-SW、10GBase-LW 和 10GBase-EW 三个规范）。这三大类规范所对应的体系结构如图 7-46 所示（注意其中用深颜色标注的部分）。

通过与图 7-44 所示的千兆以太网体系结构比较可以看出，从层次上来看，主要区别如下：

- 把原来千兆以太网的 GMII 升级替换成了 XGMII（10 Gigabit Media Independent Interface，10G 介质无关接口）。XGMII 与千兆以太网的 GMII 一样，也是位于收发器内部，但它是由全新的芯片完成的，以实现支持 10Gb/s 传输速率。XGMII 功能由一个具有 74 条信号线（相当于 74bit 宽度）的接口芯片完成，其中的 64 条数据线用于数据的收发（各 32 条），其余的为时钟和控制信号比特，是连接以太网 MAC 子层和物理层的桥梁。

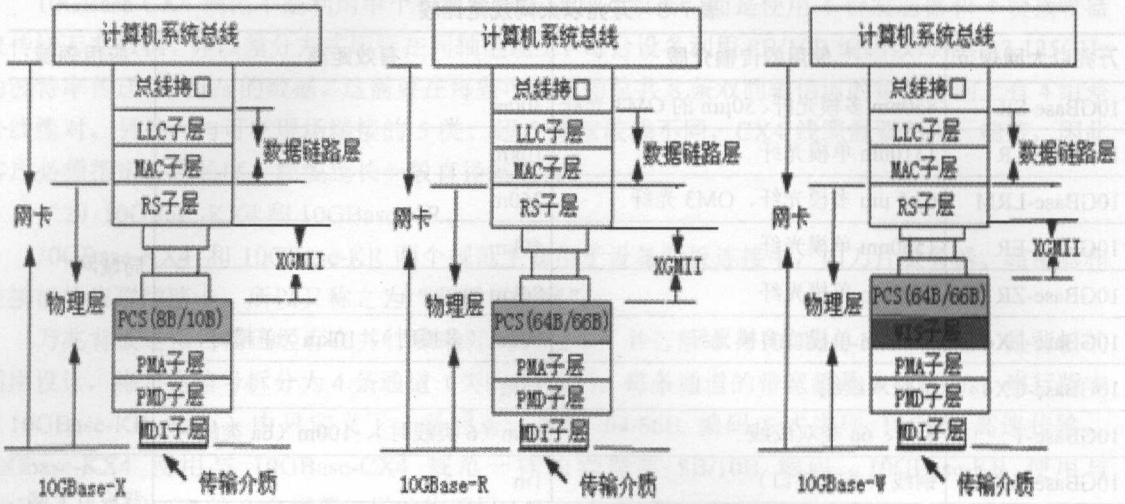


图 7-46　3 个万兆以太网规范子系列的体系结构

- 在 10GBase-R 子系列的 3 个规范中的物理层，除了上述接口换成为 XGMII 外，还有一个区别就是 PCS 子层的编码方式由原来的 8B/10B 改变成了 64B/66B。
- 在 10GBase-W 子系列的 3 个规范中相对千兆以太网物理层的改变更大，除了在 10GBase-R 子系列中的两处改变外，还在 PCS 子层与 PMA 子层之间增加了一个新的子层——WIS（WAN Interface Sublayer，WAN 接口子层）。

通过 WIS，用于广域网中产生适配 ANSI 定义的 SONET STS-192c 传输格式或 ITU 定义 SDH VC-4-64c 容器速率的以太网数据流，这就允许万兆以太网设备与同步光纤网络（SONET）STS -192c 传输格式相兼容。

其他各子层的功能参见千兆以太网体系结构中的对应子层功能说明，只不过在 PCS 子层中所采用的编码与千兆以太网不一样，参见图 7-44 所示。

【说明】随着 WLAN 技术的高速发展，接入速率超过 1G 的 WLAN 标准相继问世，如 IEEE 802.11ac 基础接入速率达到 1.3Gb/s，改进后的 IEEE 802.11ac wave2 和 IEEE 802.11ad 规范（在本章后面介绍）最高接入速率可达 7Gb/s，这显然远超当前企业网络中普遍采用的 1Gb/s 接入速率要求。尽管可以使更高的万兆以太网标准，但这样一来不仅原来普遍采用的 5 类和 5e 类需要全部替代为 6 类以上，而且 10G 设备的成本将远高于现有的千兆以太网设备，而且实际上大多数企业数据流又不需要 10G 这么高的带宽。

出于实际需要和为企业节省成本的考虑，包括像思科、华为这样的领导企业研发出了一些中间速率，如像 2.5G、5G、25G 这样的以太网标准设备。特别是像 2.5G 和 5G 这样的标准设备可以在不更新现有电缆的情况下满足绝大多数企业网络中的 IEEE 802.11ac 和 IEEE 802.11ad 无线 AP 的接入需求。

7.9 WLAN网络架构及帧格式

除了在第5章中介绍的如IEEE 802.11a/b/g/n/ac/ad这些物理层接入规范外，本节要对WLAN网络体系架构，以及MAC子层中采用的CDMA/CA（Carrier Sense Multiple Access with Collision Avoidance，载波侦听多路访问/冲突避免）协议原理和802.11 MAC帧格式进行具体介绍。下面先介绍一些有关WLAN网络体系架构的相关概念。

7.9.1 WLAN网络基本概念

WLAN广义上是指以无线电波、激光、红外线等无线信道来代替有线局域网中的部分或全部传输介质所构成的网络。本书介绍的WLAN技术是基于802.11标准系列的，即利用高频信号（如2.4GHz或5GHz或60GHz）作为传输介质的无线局域网。

802.11是IEEE在1997年为WLAN定义的一个无线网络通信的工业标准。此后这一标准又不断得到补充和完善，形成IEEE 802.11的标准系列，如802.11、802.11a、802.11b、802.11e、802.11g、802.11i、802.11n、802.1ac和802.11ad等。

在正式介绍WLAN网络体系架构前，先对一些相关基本概念进行集中介绍。

- STA（Station，工作站）：是支持802.11规范的终端设备，如带无线网卡的计算机、支持WLAN的手机等。
- 射频信号：提供基于802.11规范的WLAN技术的传输介质，是具有远距离传输能力的高频电磁波。通常所指的射频信号是2.4G或5G频段的电磁波，因为WLAN接入规范中大多数是工作在这两个频段，除了工作在60GHz频段的IEEE 802.11ad规范外。
- AP（Access Point，接入点）：为STA提供基于802.11规范的无线接入服务，起到有线网络和无线网络的桥接作用，类似于有线网络的交换机。但随着WLAN网络技术和应用的高速发展，从功能方面把AP划分成以下两类：
  - FAT AP（FAT Access Point，胖接入点）：在“自治式网络架构”（下节将介绍）中提供STA的无线接入服务，除了提供基本的无线连接功能外，还能提供安全和管理等增强功能。也就是这类AP具有全面的接入、安全和管理功能，类似于具有全面功能的交换机，以前的WLAN网络经常存在。
  - FIT AP（FIT Access Point，瘦接入点）：在“集中式网络架构”（下节将介绍）中提供STA的无线接入服务，只提供可靠、高性能的无线连接功能，其他包括安全和管理功能都交给了WLAN网络新的功能或者新的一种设备——AC（接入控制器）来集中提供。这种AP可以节省企业成本，也便于通过AC对整个网络中的许多AP设备进行集中管理。
- AC（Access Controller，接入控制器）：在集中式网络架构中，AC对无线局域网中的所有AP设备进行控制和管理。例如，AC可以通过与认证服务器交互信息来为WLAN用户提

供认证服务。目前有单独的 AC 设备，但在企业级路由器中也一般带有这种功能，担当 AC 的角色。

- CAPWAP（Control And Provisioning of Wireless Access Points，无线 AP 控制与规范）：是由 RFC5415 协议定义的，实现 AP 与 AC 之间互通的一个通用封装和传输机制。
- VAP（Virtual Access Point，虚拟接入点）：是 AP 设备上虚拟出来的业务功能实体。用户可以在一个 AP 上创建不同的 VAP 来为不同的用户群体提供无线接入服务，这样就可以实现基于用户的安全策略和 QoS 策略管理。有点类似于 VLAN 功能。
- AP 域：可以将一组 AP 划分在一个域里。域的划分由企业根据实际部署进行规划，通常是在大的 WLAN 网络中才需要配置，以便区分不同的用户类型提供不同的服务级别，部署不同的用户安全策略和 QoS 策略，管理不同类型的用户。有点类似于 MSTP（多生成树实例）中的 MST 域。
- SSID（Service Set Identifier，服务集标识符）：表示无线网络的标识，用来区分不同的无线网络。例如，当我们在笔记本电脑上搜索可接入无线网络时，显示出来的网络名称就是 SSID。根据标识方式又可以分为以下两种：
  - BSSID（Basic Service Set Identifier，基本服务集标识符）：表示 AP 上每个 VAP 的数据链路层 MAC 地址，即一个 VAP 有一个唯一的 BSSID，如图 7-47 所示。

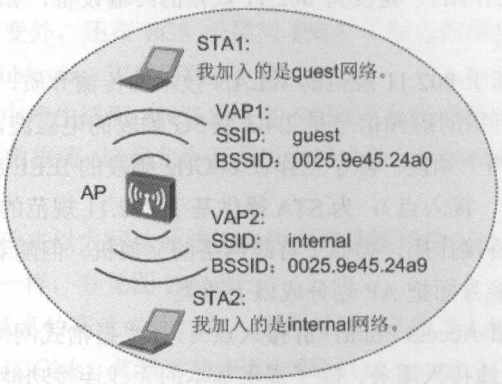


图 7-47　BSSID 示例

  - ESSID（Extended Service Set Identifier，扩展服务集标识符）：是一个或一组无线网络的标识，例如图 7-47 所示的 guest 或 internal 的 SSID 就是对应 VAP 网络的 ESSID。STA 可以先扫描所有网络，然后选择特定的 SSID 接入某个指定的无线网络。通常我们所指的 SSID 即为 ESSID。

【说明】多个 AP 可以拥有同一个 ESSID 以对用户提供漫游能力，但是对于每个 VAP 的 BSSID 必须是唯一的，因为每个 VAP 的数据链路层的 MAC 地址是唯一的。

- BSS（Basic Service Set，基本服务集）：指一个 AP 所覆盖的范围。在同一个 BSS 的服务区域内，各 STA 间是可以相互通信的。

- ESS（Extend Service Set，扩展服务集）：由多个使用相同 SSID（通常是指 ESSID）的 BSS 组成。如图 7-48 所示为 SSID、BSSID、BSS 与 ESS 的关系。

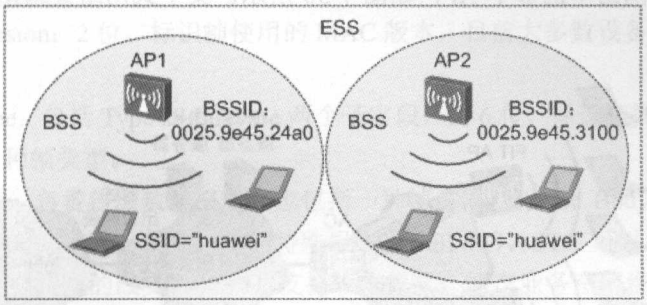


图 7-48 SSID、BSSID、BSS 与 ESS 的关系

7.9.2 WLAN 网络体系架构

WLAN 网络体系架构分有线侧和无线侧两部分：有线侧指 AP 上行连接到有线网络设备，以达到访问外部网络（如 Internet）的目的；无线侧指 STA（工作站）到 AP（接入点）之间的网络使用 802.11 协议实现网络互联的目的。

根据对网络中 AP 设备的不同管理方式，无线侧接入的 WLAN 网络体系架构分为“自治式网络架构”和“集中式网络架构”两种。

1. 自治式网络架构

自治式网络架构又称为胖接入点（FAT AP）网络架构。在该架构下，AP 可单独实现所有无线接入功能，不需要 AC 设备，如图 7-49 所示。

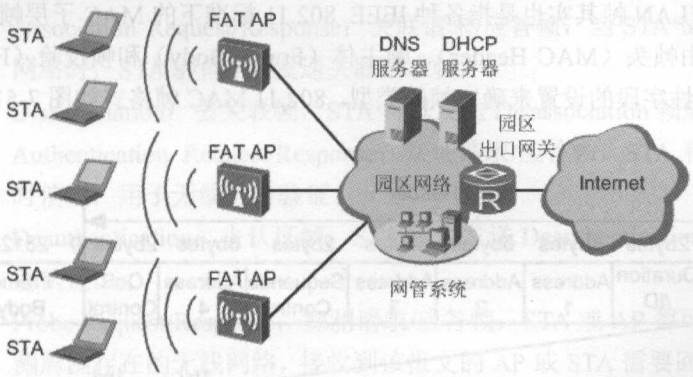


图 7-49 WLAN 自治式网络架构

WLAN 早期广泛采用自治式网络架构，各 FAT AP 自行管理自己。FAT AP 功能强大，独立性好，但是设备结构复杂，价格昂贵，难于管理。随着企业大量部署 AP，对 AP 的配置、升级软件等管理工作给用户带来很高的操作成本和维护成本，自治式网络架构应用逐步减少。

2. 集中式网络架构

集中式网络架构又称为瘦接入点（FIT AP）网络架构。在该架构下，可以通过一台 AC 集中管理和控制 WLAN 网络中的多个 AP，如图 7-50 所示。集中式网络架构便于管理员的集中管理和维护。

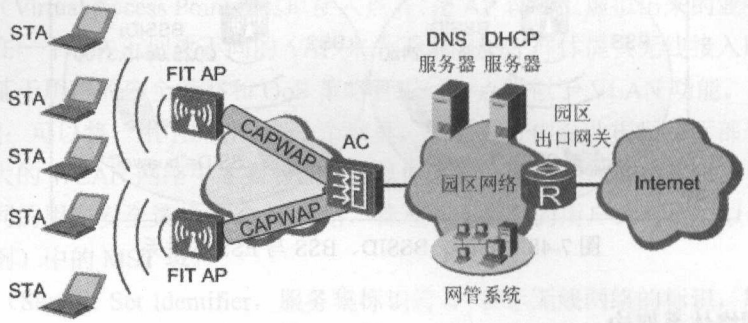


图 7-50　WLAN 集中式网络架构

在集中式网络架构下，整个无线接入功能由 AP 和 AC 共同完成：

- AC 集中处理所有的安全、控制和管理功能，如移动管理、身份验证、VLAN 划分、射频资源管理和数据包转发等。
- FIT AP 完成无线射频接入功能，如无线信号发射与探测响应、数据加密解密、数据传输确认等。
- AP 和 AC 之间采用 CAPWAP 协议进行通信，AP 与 AC 之间可以跨越二层网络或三层网络。

7.9.3　WLAN MAC 帧格式

这里所说的 WLAN 帧其实也是指各种 IEEE 802.11 标准下的 MAC 子层帧（简称 MAC 帧）。802.11 MAC 帧是由帧头（MAC Header）、帧主体（Frame Body）和帧校验（FCS）字段组成，主要依靠帧头中各属性字段的设置来确定帧的类型。802.11 MAC 帧格式如图 7-51 所示。下面对各字段进行具体介绍。

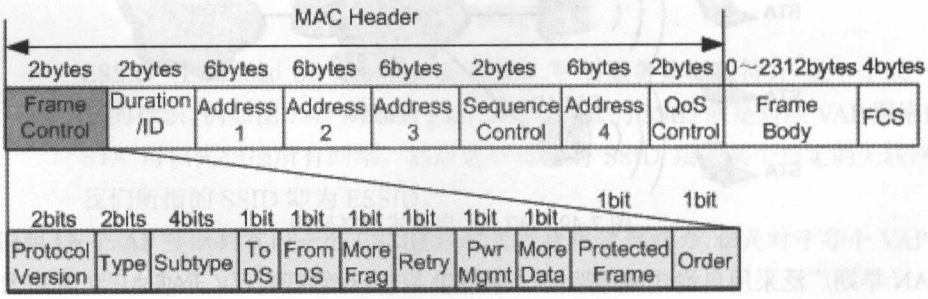


图 7-51　WLAN MAC 帧格式

1. Frame Control（FC，帧控制）字段

FC 字段占 2 个字节，用于控制 MAC 子层的帧信息和行为。在这个字段中又包括多个子字段，参见图 7-51 中的下面部分。

- Protocol Version：2 位，标识帧使用的 MAC 版本，目前大多数设备仅支持一个版本，编号为 0。
- Type/Subtype：包括 Type 和 Subtype 两个子字段，共 6 位，标识帧类型，包括：管理、控制和数据 3 种帧类型。
  - ➢ 数据帧：负责传输数据报文，也包括一种帧主体（Frame Body，也就是通常意义中的“数据”字段）部分为空的特殊报文（Null 帧）。STA 可以通过 Null 帧通知 AP 自身省电状态的改变。802.11 支持省电模式，即在业务空闲没有数据传输的情况下 STA 可以关闭天线来节省电力。
  - ➢ 控制帧：协助数据帧的传输，负责无线信道的清空、信道的获取等，还用于接收数据时的确认。常用的控制帧有：
    - ✧ ACK 帧：接收端接收报文后需要回应 ACK 帧向发送端进行确认。
    - ✧ RTS（Request To Send，请求发送）/CTS（Clear To Send，清除发送）帧：提供一种用来减少由隐藏节点问题所造成冲突的机制。发送端向接收端发送数据之前先发送 RTS 帧，接收端收到后回应 CTS 帧。通过这种机制来清空无线信道，使发送端获得发送数据的介质控制权。
  - ➢ 管理帧：负责对无线网络的管理，包括网络信息通告、加入或退出无线网络、射频管理等。常用的管理帧有：
    - ✧ Beacon：信标帧，AP 通过周期性地发送这种管理帧来宣告无线网络的存在以及支持的各类无线参数（例如 SSID、支持的速率和认证类型等）。
    - ✧ Association Request/Response：关联请求/应答帧，当 STA 试图加入到某个无线网络时，STA 会向 AP 发送关联请求帧。
    - ✧ Disassociation：去关联帧，STA 可以发送 Disassociation 帧解除与 AP 的关联。
    - ✧ Authentication Request/Response：认证请求/应答帧，STA 和 AP 进行链路认证时使用，用于无线身份验证。
    - ✧ Deauthentication：去认证帧，STA 可以发送 Deauthentication 帧解除与 AP 的链路认证。
    - ✧ Probe Request/Response：探测请求/应答帧，STA 或 AP 都可以发送探测帧来探测周围存在的无线网络，接收到该报文的 AP 或 STA 需要回应 Probe Response，Probe Response 帧中基本包含了 Beacon 帧的所有参数。
- To DS/From DS：各 1 位，标识帧是否来自/去往一个分布式系统（Distribution System，其实就是指 AP），置 1 表示帧是来自或去往 AP，否则置 0，Beacon 帧特殊，具体参见表 7-10。如果这两个标志位都置 1，则表示是 AP 到 AP 之间的帧。

- More Frag：1 位，表示是否有后续分片传送，有则置 1，无则置 0。当数据因为超过了对应 WLAN 标准链路的 MTU 后被分片时，除最后一个帧外，其他各帧在此字段值要置 1，置 0 时表示该帧为被分片后的最后一个帧或者没有经过分片的帧。类似于 IPv4 数据报格式中的 MF 标志位，参见 8.3.4 节。
- Retry：1 位，表示帧是否为重传帧，置 1 时为重传帧，正常数据发送时该位置 0，表示为非重传帧。通过此字段可以协助接收端排除重复帧，即当收到此字段置 1 的帧时，先查看本端是否已收到相同的帧（通过查看帧后面的 Sequence Number 字段值得出），如果有则丢弃（因为网络延时的原因，发送端在重传计时器超时后会重传相应的帧），否则接收。
- Pwr Mgmt：1 位，表示 STA 发送完成当前帧序列后将要进入的模式：Active（激活模式）或 Sleep（省电模式）。
- More Data：1 位，置 1 时表示 AP 向省电状态的 STA 传送缓存报文。
- Protected Frame：1 位，表示当前帧是否已经被加密，已经过加密置 1，否则置 0。
- Order：1 位，置 1 时表示帧是按顺序传输的。当有些帧优先级较高时，则打乱原来的帧发送顺序，则此时会置 0。

2. 其他字段

介绍完 FC 中各字段的含义后，下面再来介绍图 7-51 中的其他字段。

（1）Duration/ID 字段。

2 个字节，根据填充值的不同，其作用包括：

- 实现 CSMA/CA 的网络分配向量机制，表示 STA 占用信道的时间，即信道处于忙状态的持续时间，即 NAV（网络分配向量）值。
- 无竞争周期所传送的帧：填充值为 32768 时，表示一直占用信道，其他 STA 不能竞争。
- PS-Poll 帧：省电一轮询帧，STA 的工作模式包括激活模式（Active）和省电模式（Sleep）。STA 进入省电模式后，AP 会缓存到此 STA 的数据帧。当 STA 从省电模式切换到激活模式时，STA 可以向 AP 发送 PS-Poll 帧来获取缓存帧。此时，PS-Poll 帧的该字段包含关联标识符 AID（Association ID），用来标识 STA 所属的 BSS。

（2）Address n 字段。

标识 4 个 MAC 地址，各 6 个字节。但这 4 个 MAC 地址的填法不固定，需要与 Frame Control 字段中的 To DS/From DS 标志位结合来确定。例如，帧从一个 STA 发往 AP 与从 AP 发往 STA，4 个 Address 字段的填法是不一样的。Address n 字段填写规则如表 7-10 所示。

表 7-10　Address n 字段填写规则

To DS	From DS	Address 1	Address 2	Address 3	Address 4	说明
0	0	目的 MAC 地址	源 MAC 地址	BSSID	未使用	管理帧与控制帧，例如 AP 发送的 Beacon 帧
0	1	目的 MAC 地址	BSSID	源 MAC 地址	未使用	如图 7-52 中的(1)，AP1 向 STA1 发送的帧

续表

To DS	From DS	Address 1	Address 2	Address 3	Address 4	说明
1	0	BSSID	源 MAC 地址	目的 MAC 地址	未使用	如图 7-52 中的(2)，STA2 向 AP1 发送的帧
1	1	目的AP的BSSID	源 AP 的BSSID	目的 MAC 地址	源 MAC 地址	如图 7-52 中的(3)，AP1 向 AP2 发送的帧

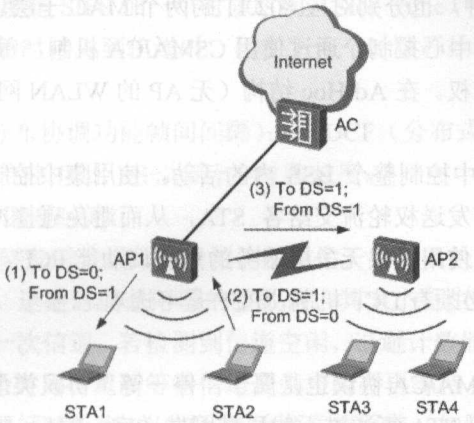


图 7-52 4 个 MAC 地址字段填写的网络参考示例

（3）Sequence Control 字段。

序列控制字段，2 个字节，用来丢弃重复帧和重组分片，包含两个子字段：

- Fragment Number：分片号，用于标识分片帧，在帧被分片时才赋值，在接收端也可根据本字段对由同一个帧拆分的各分片按分片号大小顺序进行重组。
- Sequence Number：帧序列号，用于检验重复帧，由同一帧拆分的各分片中的本字段值是一样的，用于在接收端实现帧重组。当设备收到一个 802.11 MAC 帧，其 Sequence Number 与之前收到的帧重复，则丢弃该帧。

（4）QoS Control 字段。

2 个字节，只存在于数据帧中，用来实现基于 802.11e 标准的 WLAN QoS 功能。

（5）Frame Body 字段。

也称为“数据”字段，0～2312 个字节，负责传输上层的有效载荷（Payload）。在 802.11 标准中，传输的载荷报文也被称为 MSDU（MAC Service Data Unit，MAC 服务数据单元）。

（6）FCS（Frame Check Sequence，帧校验序列）字段。

4 个字节，用于检查接收帧的完整性。类似于以太网中的 CRC。

7.9.4 CSMA/CA 协议原理

在有线局域网中，IEEE 802.3 标准使用了 CSMA/CD（载波侦听多址访问/冲突检测）机制来协

调不同的设备对有线介质的竞争访问。CSMA/CD 机制的基础是所有站点都能“听见”（监听）对方，而无线局域网中的无线信号覆盖范围有限，并非所有的站点都能够听见对方。因此，802.11 标准提出了载波侦听多址访问/冲突避免 CSMA/CA 机制。

IEEE 802.11 MAC 层是通过协调功能（Coordination Function）来确定 BSS（基本服务集，标识一个 AP）中的 STA 之间如何发送或接收数据。根据所采用的介质争用解决方法的不同，协调功能又分为 DCF（Distributed Coordination Function，分布式协调功能）和 PCF（Point Coordination Function，点协调功能）两种，也分别对应 802.11 的两个 MAC 子层。

- DCF：不采用任何中心控制，通过使用 CSMA/CA 机制，每个 STA 均可通过争用信道来获取数据帧的发送权。在 Ad-Hoc 结构（无 AP 的 WLAN 网络）的 WLAN 网络中就是采用 DCF 功能。
- PCF：通过 AP 集中控制整个 BSS 内的活动，使用集中控制的接入算法，用类似于探询的方法把数据帧的发送权轮流交给各 STA，从而避免碰撞冲突。对于时间敏感的业务，如分组话音，就应使用提供无争用服务的点协调功能 PCF。

在 IEEE 802.11 协议中必须有 DCF 机制，PCF 是可选项。

1. 基本概念

IEEE 802.11 使用的 CSMA/CA 协议也是属于“停一等”协议类型。这是因为无线信道的通信质量远不如有线信道的，所以 STA 每通过无线局域网发送完一帧后，要等到收到对方的确认（ACK）帧后才能发送下一帧。这叫做链路层确认。

CSMA/CA 协议的工作原理可用图 7-53 描述，下面先介绍相关的基本概念。

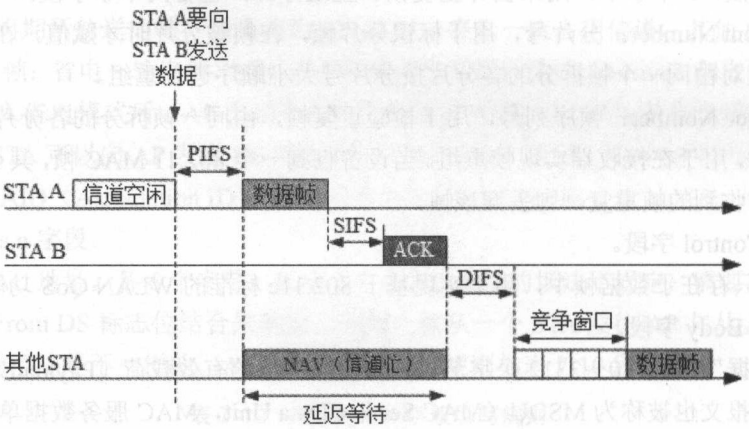


图 7-53　CSMA/CA 工作原理示意图

（1）IFS（InterFrame Space，帧间间隔）。

IEEE 802.11 规定，所有 STA 完成数据帧发送后，必须等待 IFS 时间才能发送下一帧。帧间间隔的长短取决于该站要发送的帧的类型。高优先级帧需要等待的时间较短，可以优先获得发送权。

常用的 3 种帧间隔如下：

- SIFS（Short IFS，短帧间间隔）：SIFS 是最短的帧间间隔，用来分隔属于一次对话的各个帧。在这段时间内，一个 STA 应当能够从发送方式切换到接收方式。使用 SIFS 的帧类型有：ACK 帧、CTS 帧、由过长的 MAC 帧分片后的数据帧，以及所有回答 AP 探询的帧和在 PCF 方式中接入点 AP 发出的任何帧。
- PIFS（PCF IFS，点协调功能帧间间隔）：在 PCF（点协调功能）中实现优先获得接入介质功能。PIFS 的长度是 SIFS 加一个时隙时间（Slot Time）长度。在一个 BSS 内，当某 STA 在一个时隙开始时接入到信道时，则在下一个时隙开始时，其他 STA 都能检测出信道处于忙状态。
- DIFS（DCF IFS，分布协调功能帧间间隔）：在 DCF（分布式协调功能）中用来发送数据帧和管理帧。DIFS 的长度比 PIFS 多一个时隙长度。

（2）竞争窗口：也叫做退避时间。

如果同时有多个 STA 需要发送数据，都检测到信道忙，则需要执行退避算法，即每个 STA 随机退避一段时间再发送数据。退避时间是时隙的整数倍，其大小是由物理层技术决定的。STA 每经历一个时隙的时间就检测一次信道。若检测到信道空闲，退避计数器继续倒计时；若检测到信道忙，就冻结退避计时器的剩余时间，重新等待信道变为空闲并再等待 DIFS 后从剩余时间开始继续倒计时，直到退避计时器的时间减小到 0，STA 开始发送数据帧，如图 7-54 所示。

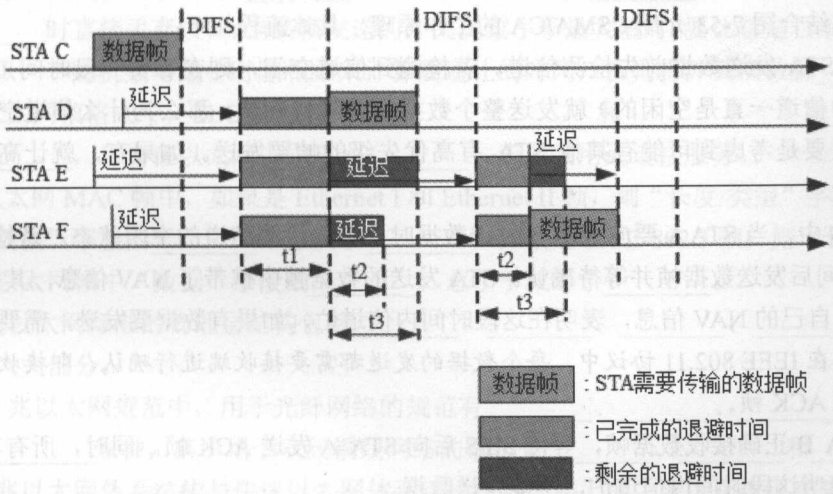


图 7-54 退避原理示意图

具体流程描述如下：

①根据图 7-54 可知，STA C 正在占用无线信道发送数据帧，此时，假设 STA D、STA E 和 STA F 也想发送数据，但检测到无线信道忙，开始等待延迟发送。

②待 STA C 的数据帧发送完成后，所有 STA 需要等待 DIFS 时间。DIFS 帧间隔时间后，需要发送数据帧的 STA 会随机生成一个退避时间并启动退避计时器。假设 STA D 的退避时间为 t1，STA E 的退避时间为 t1+t3，STA F 的退避时间为 t1+t2。

③t1 时间后，STA D 退避计时器最先减小到 0，开始发送数据。

④此时，STA E 和 STA F 又会检测到无线信道忙，需要再次停止退避，冻结退避计时器的剩余时间并继续等待，直到 STA D 的数据帧传输完成后再等待一个 DIFS 时间后，STA E 和 STA F 的退避计时器继续倒计时。

⑤当 t2 时间到后，STA F 退避计时器最先减小到 0，STA F 开始发送数据帧。

2. CSMA/CA 原理

在正式介绍 CSMA/CA 工作原理前，首先需要了解，为了尽量减少碰撞的机会，IEEE 802.11 标准采用了一种叫做 VCS（Visual Carrier Sense，虚拟载波监听）的机制，让源 STA 把它要占用信道的时间（包括目的站点发回确认帧所需的时间）以微秒为单位写入到所发送的 MAC 帧头部中的 Duration/ID（标识“持续时间”）字段中，一直到目的 STA 把 ACK 确认帧发送完为止，以便使其他所有站在这一段时间都不要发送数据。

当 STA 监测到正在信道中传送的帧中带有“持续时间”字段值时，就同步调整自己的 NAV（Network Allocation Vector，网络分配向量）。NAV 指出了信道处于忙状态的持续时间。信道处于忙状态就表示：或者是由于物理层的载波监听检测到信道忙，或者是由于 MAC 层的虚拟载波监听机制指出了信道处于忙状态，不能发送数据。

下面正式结合图 7-53 介绍 CSMA/CA 的工作原理，具体流程如下：

（1）在 STA 发送数据前先检测信道。若检测到信道空闲，则在等待一段时间 DIFS 后（如果这段时间内信道一直是空闲的）就发送整个数据帧并等待确认。那么为什么信道空闲时还要再等待呢？这主要是考虑到可能有其他 STA 有高优先级的帧要发送。如果有，就让高优先级的帧先发送。

在图 7-53 中，当 STA A 要向 STA B 发送数据时，首先检测信道的空闲状态。若检测到空闲，等待 DIFS 时间后发送数据帧并等待确认。STA 发送的数据帧中携带了 NAV 信息，其他 STA 接收到此帧后更新自己的 NAV 信息，表明在这段时间内信道忙，如果有数据要发送，需要延迟等待。

【说明】在 IEEE 802.11 协议中，每个数据的发送都需要接收端进行确认，即接收端收到数据向发送端发送 ACK 帧。

（2）STA B 正确接收数据帧，等待 SIFS 后向 STA A 发送 ACK 帧。同时，所有其他 STA 都设置 NAV，表明这段时间内信道忙，不能发送数据。

（3）当 ACK 帧发送结束时，信道开始空闲。等待 DIFS 时间后，接着会出现一段空闲时间，叫做争用窗口，表示在这段时间内有可能出现各 STA 争用信道的情况。此时，需要发送数据的各 STA 开始利用退避算法争用信道。退避计数器最先减小到 0 的 STA 开始发送数据帧。

IEEE 802.11 CSMA/CA 中使用的退避算法和以太网 CSMA/CD 中使用的退避算法稍有不同。在 CSMA/CA 中，第 i 次退避是在时隙{0,1,2,...,$(2^{(2+i)}-1)$}中随机地选择一个退避时间，当时隙编号

达到255时（这对应于第6次退避）就不再增加了。这样做是为了使不同站点选择相同退避时间的概率减少。

7.10 课后自测题

为了巩固本章所学的内容，请认真做下面的自测题，参考答案见附录。

7.10.1 填空题

1．以太网MAC帧分为3种，当目的MAC地址中的最高位为1时是_____，最高位为0时为_____，所有位均为1时为_____。

2．在局域网中常见的介质访问控制协议有3种，其中用于总线型网络的是_____，用于以太网的是_____，用于WLAN网络的是_____。

3．CSMA所采用的退避算法有3种，其中一直连续监听介质状态的是_____，不连续监听介质状态的是_____，可不一直连续监听介质状态的是_____。

4．CSMA/CD是CSMA的改进版，增加了_____和_____功能，它的功能原理可用以下3句话来描述：_____、_____和_____。其中的“听”是指_____，其中的“说”是指_____。

5．CSMA/CD的退避算法为_____，退避的最小时隙为_____，且规定当某帧发生冲突数达到_____时直接丢弃该帧，当某帧发送的冲突次数小于最大值时，要比较是否小于_____，如果是，则以一个随机数与最小时隙相乘来计算要避让的时间，否则将冲突次数置_____。

6．在以太网帧格式中，LLC头部包括_____、_____和_____3个字段，其中前两者固定值为_____，后者在以太网中固定为_____，表示采用无连接服务。

7．在以太网MAC帧中，如果是Ethernet I和Ethernet II帧，则“长度/类型”字段表示的是以太网帧中“数据”字段的_____，总大于_____，如果是IEEE 802.3帧，则“长度/类型”字段表示的是以太网帧中“数据”字段的_____，总小于或等于_____。

8．快速以太网标准的物理层结构包括_____、_____、_____、_____、_____和_____六大部分。

9．在千兆以太网规范中，用于光纤网络的规范有_____、_____、_____、_____、_____和_____6种，其中用于双绞铜线网络的规范是_____、_____和_____3种。

10．千兆以太网体系结构与快速以太网体系结构在物理层上的主要区别表现在RS子层和PCS子层之间的接口发生了改变，即由快速以太网中的_____接口扩展为_____接口。另外，1000Base-CX、1000Base-LX和1000Base-SX三个IEEE规范采用的是_____编/解码方式，1000Base-T规范采用的是_____编/解码方式，快速以太网规范采用的是_____编/解码方式。

11．10G以太网规范非常多，其中用于局域网的万兆光纤以太网规范有_____、_____、_____、_____、_____和_____6个，用于局域网的万兆双绞（或铜）线以太网规范有

_____、_____、_____和_____4 个，用于广域网的万兆光纤以太网规范有_____、_____、_____和_____4 个。IEEE 发布的万兆以太网规范又可分为三大类，即_____、_____和_____。

12. WLAN 网络体系架构总体分为两种，其中适用于 FAT AP 的称为_____，适用于 FIT AP 的称为_____。

13. 在 WLAN 通信中，发送端向接收端发送数据之前先发送_____帧，接收端收到后回应_____帧，以此来清空无线信道，使发送端获得发送数据的介质控制权。AP 还会周期性地发送_____帧来宣告无线网络的存在。

7.10.2 选择题（可多选）

1. 在 CSMA 所用的介质避让算法中，介质利用率最高的是（　　）。

A. 非-坚持 CSMA　　B. 1-坚持 CSMA

C. p-坚持 CSMA　　D. 截断二进制指数退避算法

2. 在 IEEE 802 系列局域网标准中，用来定义 LLC 子层的是（　　）。

A. IEEE 802.1　　B. IEEE 802.2　　C. IEEE 802.3

D. IEEE 802.4　　E. IEEE 802.6

3. 在 IEEE 802 系列局域网标准中，用来定义 CSMA/CD 协议的是（　　）。

A. IEEE 802.1　　B. IEEE 802.2　　C. IEEE 802.3

D. IEEE 802.4　　E. IEEE 802.6

4. 在 IEEE 802 系列局域网标准中，可用于城域网（MAN）的是（　　）。

A. IEEE 802.1　　B. IEEE 802.2　　C. IEEE 802.3

D. IEEE 802.4　　E. IEEE 802.6

5. 下列标准中（　　）是 IEEE 发布的第一个以太网帧格式版本。

A. Ethernet II　　B. Ethernet 802.3 raw

C. Ethernet 802.3 SAP　　D. 802.3/802.2 LLC

6. 下列标准中（　　）专门适用于 Novell 以太网帧。

A. Ethernet II　　B. Ethernet 802.3 raw

C. Ethernet 802.3 SAP　　D. 802.3/802.2 LLC

7. 下列以太网标准中不包括 LLC 帧头部分内容的是（　　）。

A. Ethernet II　　B. Ethernet 802.3 raw

C. Ethernet 802.3 SAP　　D. 802.3/802.2 LLC

8. 在 Ethernet 802.3 raw 帧中 Data 字段的最大字节数为（　　）。

A. 1500　　B. 1498　　C. 1497　　D. 1492

9. 在 802.3/802.2 SNAP 帧中 Data 字段的最大字节数为（　　）。

A. 1500　　B. 1498　　C. 1497　　D. 1492

10. 下列以太网帧格式中把 Ethernet II 帧头的 Type 字段替换为帧 Length 字段的是（ ）。

A. Ethernet 802.3 raw　　B. Ethernet 802.3 SAP

C. 802.3/802.2 LLC　　D. 802.3/802.2 SNAP

11. 下列以太网 MAC 帧 Data 字段内容仅包括来自网络层的数据包的是（ ）。

A. Ethernet II　　B. Ethernet 802.3 raw　　C. Ethernet 802.3 SAP

D. 802.3/802.2 LLC　　E. 802.3/802.2 SNAP

12. 以太网帧的最小长度和最大长度（均以字节为单位）分别为（ ）。

A. 43　1497　　B. 44　1498　　C. 64　1500　　D. 64　1518

13. 在 802.3/802.2 SNAP 帧格式中，如果上层协议是 IPv4 协议，则在“类型”字段中对应的值为（ ）。

A. 0x0800　　B. 0x0806　　C. 0x8138　　D. 0x86DD

14. 在以太网 MAC 帧中“帧起始”字段的值为固定的（ ）。

A. 01111110　　B. 11111111　　C. 10101011　　D. 10101010

15. 整体而言，在所有以太网 MAC 帧中，“数据”字段的字节数取值范围为（ ）。

A. 38～1500　　B. 38～1518　　C. 64～1500　　D. 64～1518

16. 在以下标准以太网规范中使用双绞线作为传输介质的是（ ）。

A. 10Base2　　B. 10Base5　　C. 10Broad36　　D. 10Base-T

17. 下列标准光纤网规范中，主要用于光纤中继器间连接的是（ ）。

A. 10Base-F　　B. 10Base-FB　　C. 10Base-FL　　D. 10Base-FP

18. 下列标准以太网物理层结构中（ ）是直接与传输介质连接的。

A. MAU　　B. MAC　　C. AUI　　D. PLS

19. 下列标准以太网物理层结构中（ ）定义了传输介质的机械和电气特性。

A. MAU　　B. MAC　　C. AUI　　D. PLS

20. 下列快速以太网规范中必须要同时使用双绞线 4 对芯线的是（ ）。

A. 100Base-T　　B. 100Base-TX　　C. 100Base-FX　　D. 100Base-T4

21. 相对标准以太网标准物理层结构，在快速以太网标准物理层结构中属于新增子层的是（ ）。

A. MDI 子层　　B. PMD 子层　　C. PMA 子层

D. PCS 子层　　E. RS 子层

22. 下列光纤千兆以太网规范中，是 IEEE 以标准形式发布的是（ ）。

A. 1000Base-LX　　B. 1000Base-SX　　C. 1000Base-LH

D. 1000Base-ZX　　E. 1000Base-LX10

23. 下列千兆以太网规范中只能采用多模光纤的是（ ）。

A. 1000Base-LX　　B. 1000Base-SX　　C. 1000Base-LH

D. 1000Base-ZX　　E. 1000Base-LX10　　F. 1000Base-BX10

24. 下列千兆以太网规范中只能采用单模光纤的是（　　）。

A. 1000Base-LX　　B. 1000Base-SX　　C. 1000Base-LH

D. 1000Base-ZX　　E. 1000Base-LX10　　F. 1000Base-BX10

25. 下列千兆以太网规范中既可采用单模光纤，又可采用多模光纤的是（　　）。

A. 1000Base-LX　　B. 1000Base-SX　　C. 1000Base-LH

D. 1000Base-ZX　　E. 1000Base-LX10　　F. 1000Base-BX10

26. 下列千兆以太网规范中采用 150Ω 平衡屏蔽双绞线（STP）作为传输介质的是（　　）。

A. 1000Base-SX　　B. 1000Base-CX　　C. 1000Base-T　D. 1000Base-TX

27. 下列千兆以太网规范中只能采用 6 类或 7 类双绞线的是（　　）。

A. 1000Base-SX　　B. 1000Base-CX　　C. 1000Base-T　D. 1000Base-TX

28. 在千兆以太网规范中有效传输距离最大的是（　　）。

A. 1000Base-LX　　B. 1000Base-SX　　C. 1000Base-LH

D. 1000Base-ZX　　E. 1000Base-LX10　　F. 1000Base-BX10

29. 下列千兆以太网规范中不是采用 8B/10B 编/解码方式的是（　　）。

A. 1000Base-T　　B. 1000Base-LX　　C. 1000Base-SX

D. 1000Base-TX　　E. 1000Base-CX

30. 下列万兆以太网规范中仅支持多模光纤的是（　　）。

A. 10GBase-SR　　B. 10GBase-LR　　C. 10GBase-LRM

D. 10GBase-SW　　E. 10GBase-EW

31. 下列万兆以太网规范中同时支持单模和多模光纤的是（　　）。

A. 10GBase-SR　　B. 10GBase-LR　　C. 10GBase-LRM

D. 10GBase-SW　　E. 10GBase-LX4

32. 下列万兆以太网规范中不是应用于局域网的是（　　）。

A. 10GBase-SR　　B. 10GBase-LW　　C. 10GBase-LR

D. 10GBase-ER　　E. 10GBase-LX4

33. 下列万兆以太网规范中不是由 IEEE 发布的是（　　）。

A. 10GBase-SW　　B. 10GBase-LW　　C. 10GBase-EW

D. 10GBase-ZW　　E. 10GBase-LRM　　F. 10GBase-ZR

34. 在 10GBase-W 子系列的 3 个规范中相对千兆以太网物理层新增了一个（　　）。

A. XGMII 接口　　B. WIS 子层　　C. PCS 子层　　D. PMA 子层

35. 在 IEEE 发布的万兆以太网规范中采用的编/解码方式是（　　）。

A. 4B/5B　　B. 8B/10B　　C. PAM-5　　D. 64B/66B

36. 我们平常使用的 VLAN 技术所对应的协议是（　　）。

A. IEEE 802.1d　　B. IEEE 802.1w　　C. IEEE 802.s　　D. IEEE 802.1q

37. 下面关于 SSID、BSSID、BSS 与 ESS 关系说法正确的是（　　）。

A．ESS 包括 BSS，但每个 BSS 的 SSID 必须不同

B．一个 AP 只能配置一个 SSID，但却可以配置多个 BSSID

C．不同 BSS 中配置的 SSID 可以相同也可以不同

D．一个 BSS 只能配置一个 BSSID，一个 ESS 只能配置一个 SSID

38．在 WLAN 中，当 To DS 和 From DS 两个标志位分别置 1 和 0 时代表的帧是（　　）。

A．AP 发送的 Beacon 帧　　B．AP 向 STA 发送的帧

C．STA 向 AP 发送的帧　　D．AP 向 AP 发送的帧

39 如果某 WLAN MAC 帧被分片了，在第一个分片中（　　）标志位将必须置 1。

A．More Frag　　B．More Data　　C．Order　　D．Retry

40 下面关于 WLAN 网络数据发送说法不正确的是（　　）。

A．STA 每发送完一个数据帧后，要等到收到对方的确认（ACK）帧后才能发送下一帧

B．STA 完成数据帧发送后，必须等待 SIFS 时间才能发送下一帧

C．STA 在发送数据前，每经历一个时隙的时间就检测一次信道的状态

D．在 STA 检测到信道忙时，退避计时器继续计时，当退避计时器的时间减小到 0 时开始发送数据帧

8 网络层

如果我们把“物理层”和“数据链路层”比作构建市内交通的话，那么本章所要介绍的“网络层”就可以比作连接不同城市交通的中转车站、机场或码头。就像这些中转车站、机场或码头可以把来自其他城市的旅客转送到下一站或本市目的地一样，网络层可以把来自其他网络中的数据转送到下一个途经的网络或本网络中的目的节点。当源端和目标端位于不同网络的时候，直接通信就不可行，就需要由网络层解决。

“网络层”是网络体系结构中非常重要的一层，在技术上又是非常复杂的一层，因为它既要解决不同网络节点间通信的路由和协议识别问题，又要通过路由选择策略解决网络拥塞问题，尽可能提高网络通信的可靠性。网络层关注的是如何将分组从源端沿着网络路径传送到目标端。为了实现这个目标，网络层必须知道通信子网的拓扑结构，并且在拓扑结构中选择适当的数据转发路径。同时网络层还必须谨慎地选择数据转发路径，以避免发生某些通信线路和路由器负载过重，而其他线路和路由器空闲的情形。

本章所介绍的内容非常丰富，对理解计算机网络通信原理和网络故障排除方法也非常重要。主要内容包括：网络层服务类型、主要路由算法、拥塞控制方法及原理、主要网络层通信协议报文格式和工作原理、三层交换原理。关于 IP 地址及子网划分将第 9 章介绍。

本章重点与难点：

- 报文交换与分组交换原理
- 虚电路分组交换和数据报分组交换原理
- IPv4 和 IPv6 报文格式
- IPv4 数据报分段与重组原理
- ARP 协议报文格式和地址解析原理

- 免费 ARP、代理 ARP 和 RARP 工作原理
- ICMP 报文格式和在 Ping 与 Tracert 中的应用
- 路由表基础和路由优先级
- 最短路径路由算法、距离矢量路由算法和链路状态路由算法原理
- 网络拥塞控制方法及原理
- 三层交换原理

8.1 网络层概述

本书前面介绍的“物理层”和“数据链路层”构建了局域网内部的通信线路，相当于一个城市内部的交通路线，本章所要介绍的“网络层”就是用来连接不同局域网线路的结点，位于不同网络的边缘，就像各城市边缘用于连接不同城市交通线路的中转站点一样。

8.1.1 划分网络层的必要性

“网络层”是从功能上定义的一个逻辑层次，主要负责不同网络间的互联，以及数据在不同网络间转发时的路径选择（即网络间的寻址）。与“物理层”和“数据链路层”有具体的设备支持一样，“网络层”也是有具体的设备来完成其相关任务，最典型的就是我们常用的路由器（包括具有三层功能的三层交换机）。路由器就相当于用来连接不同城市公路的中转车站，起数据中转作用，如图 8-1 所示。每个路由器至少可连接两个网络，就像一个城市的中转车站可以连接多个城市的交通一样。

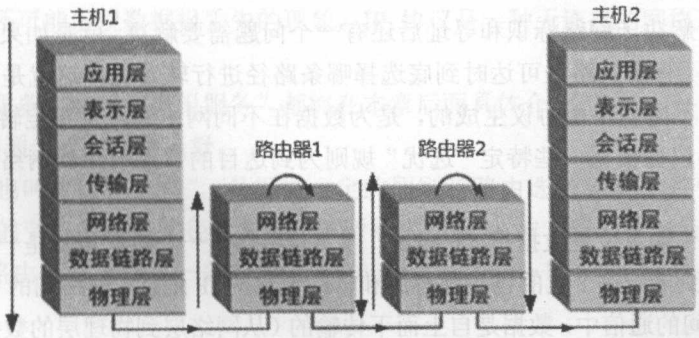


图 8-1 不同网络的连接示例

网络层的产生也是计算机网络发展的产物，并不是有了计算机网络就有了它。在早期的计算机网络中，基本上是以独立的局域网形式存在。随着计算机网络的普及和发展，人们越来越发现，非常有必要把这些一个个孤立的局域网连接起来，组成一个更大的计算机网络，这样计算机网络的作用更能

显现，可使更多人共享服务器和硬件资源。这就涉及到不同计算机网络间的互联问题了。另外，如果发送端和目的端中间隔了好几个网络，而且可达的路径有多条时，数据转发该选择哪条路径呢？就像我们旅游到某个景点，有多条路线时，该选择哪条路线一样。这都是网络层要解决的问题。

那为什么在体系结构中要有这样一个“网络层”呢？主要原因体现在以下 3 个方面：

（1）不同类型网络的转换。

由于不同的计算机网络运行的网络层通信协议可能不同，对应的网络层地址类型肯定不同，这时就需要有负责协议和地址转换的功能，这就是网络层的一个重要作用，网络层设备通常是支持多种不同网络层通信协议的。当然，其实在当前大多数网络都是 IP 网络，所以网络层通信协议和地址类型都是一样的。

（2）网络寻址。

即使是同种类型的计算机网络间互联，也不能直接拉一条电缆就可以把不同网络成功连接起来，因为这同样涉及到寻址的问题，当然这里进行的是网络层的寻址，在 IP 网络中就是 IP 地址的寻址。在不同计算机网络间的通信中，源节点要确定目的节点是在哪个网络中必须通过三层地址来寻址。这就要求在不同网络中有统一的网络标识规范，就像我们通过邮局寄信件都要填上收件人所在城市的邮政编码（这是由邮局统一编码的）一样，这个邮政编码就相当于计算机网络中的 IP 地址。

当数据转发到了目的网络后就不再按 IP 地址来寻址了，而是按照仅在同一网段有效的数据链路层 MAC 地址来继续查找真正的目的节点。就像当我们邮寄的信件到达了收件人所在的城市后就不能再按邮政编码来寻址了，而是要根据信件上留的具体街道地址来找了。

通过以上分析可以知道，计算机网络的寻址也是分层的，就像我们旅游一样，先坐城市间，甚至国家间可用的交通工具到达旅游景点所在的城市，然后再在市内坐市内交通工具到对应的旅游景点。

（3）网络路由。

通过三层地址解决了网络标识和寻址后还有一个问题需要解决，就是如果源网络和目的网络中间隔了多个网络，有多条路径可达时到底选择哪条路径进行转发呢？这就是我们常说的“路由”问题了。路由是由不同的路由协议生成的，是为数据在不同网络间转发指定转发路径的，数据在转发时这些路由协议会根据一些特定“选优”规则为到达目的节点所在的网络选择一条最优的路由路径。

“网络层”的传输单位是数据报（Packet，或“分组”），一个数据报就是一个数据帧经过网络层协议重封装后得到的，每个包的包头都有源和目的节点的 IP 地址，网络间的寻址就是通过 IP 地址进行的。在网络间的通信中，数据是自上而下传输的（从网络层到物理层的数据单位依次是“包”“帧”和“比特”），接收方是自下而上传输的，如图 8-2 所示。

8.1.2 网络层的主要作用

“网络层”为它的上一层——“传输层”提供服务，并接受它的下一层——“数据链路层”所提供的服务。网络层的主要作用表现在以下几个方面：

（1）屏蔽网络差异，提供透明传输。

因为不同网络有不同的规范要求，就像不同城市有不同的交通规范一样。网络层就是为了解决这种不同差异，寻找一个不同网络间能共同遵守的网络通信规范，以便不同网络间能相互识别并接受对方的访问请求。这样传输层就可以在不同网络间进行透明（也就是不管不同网络间的差异，就像通信双方是直接互连一样）数据传输。

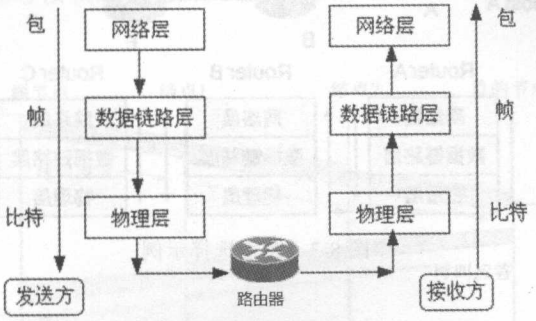


图 8-2 不同网络间主机通信的各层数据传输单位

“网络层”向“传输层”提供的服务有两类：面向连接的网络服务和无连接的网络服务。“虚电路服务”是网络层向传输层提供的一种面向连接的网络服务，是可使所有数据报按顺序到达目的结点的可靠数据传送方式。在虚电路服务中，进行数据交换的两个结点之间存在着一条专为它们服务的虚电路（相当于专线连接或者点对点连接），无需附加网络地址。如我们所使用的各种拨号 WAN 连接、数据专线等接入方式都是属于面向连接的虚电路服务方式。

“数据报服务”是网络层提供的一种无连接的网络服务，只能提供不可靠的数据传送方式，源结点（如源路由器）发送的每个数据报都要附加网络地址、包序号等信息；目的结点收到的数据报不一定按序到达，还可能出现数据报丢失的现象。IP 协议是一种无连接的网络层协议，提供无连接的网络层服务。

有关“虚电路服务”和“数据报服务”都将在本章后面具体介绍。

（2）为网络间通信提供路由选择。

“路由选择”也叫“路径选择”，是根据一定的原则和路由选择算法在多个结点的通信子网中选择一条到达目的节点的最佳路径。如图 8-3 所示，Host A 与 Host B 之间可以有多条通信路径选择，最终根据路由算法选择了一条 Host A→Router A→Router B→Router C→Host B 的路由转发路径。

在无连接的数据报服务中，网络结点设备要为每个数据报作出数据下步转发的路由选择，就像你到达另一个城市时并不清楚该市的交通路线，所以只能在中转车站查看对应的交通线路来查找一个到达你要去的目的地的最佳线路；而在面向连接的虚电路服务中，因为在建立连接时就已确定了路由路径，就像你坐专车直达目的地一样，根本无需选择旅行的路线。

（3）数据报封装和解封装。

网络层同样面临数据报封装和解封装的问题，即在发送端对来自传输层的数据段加装网络层协议头部，在接收端对需要向上面的传输层传输的数据去掉数据报中的网络层协议头。

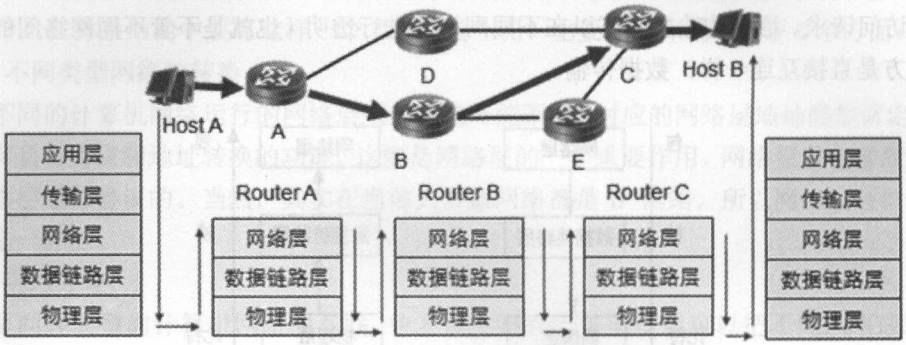


图 8-3　路由选择示例

（4）拥塞控制。

拥塞控制是为了避免网络传输路径中数据的传输延迟或死锁。在“数据链路层”我们提到了“流量控制”功能，那是针对数据链路中点对点传输速率的控制，而这里的“拥塞控制”是针对网络传输路径中不同网络间的数据传输控制，使不同网络间的数据传输速率尽可能匹配，以免造成数据溢出。在“网络层”进行拥塞控制主要采用预约缓冲区、许可证和分组丢弃等方法，具体将在本章后面介绍。

8.2　网络层数据交换及相关技术

数据到了网络层后，路由器（或三层交换机）是如何把这些数据报（或“数据分组”）转发到位于另一网络中的目的节点的呢？这就涉及到网络层的数据交换技术了。从整个网络层数据交换技术的发展来看，经历了一个由“线路交换”到“报文交换”再到现在最常用的“分组交换”的历程。“报文交换”和“分组交换”都属于“存储－转发”交换技术。

在计算机网络中，两个端点之间通常需要通过中间结点实现数据通信，这些中间结点并不关心数据内容，而只是提供一个交换设备，把数据从一个结点转发到另一个结点，直至达到目的端。数据交换技术主要是指网络中间结点所提供的数据交换功能。

8.2.1　线路交换原理

“线路交换”（Circuit Switching，也称“电路交换”）是最原始的数据交换方式，主要应用于早期的电话交换网（PSTN）和 ISDN（Integrated Services Digital Network，综合业务数字网）B 通道。在这种交换方式中，任意两个节点间要进行通信，必须先在这两个节点间通过网络建立专用的通信线路（或通道），而且这条线路会占用整个通信通道的带宽，并在两节点通信期间保持连接。

另外，在线路交换方式中，整个通信数据都是一次性传输的，即不像本章后面将要介绍的“分组交换”那样要把一个大数据拆分成多个小的分组。

在整个线路交换的数据传输过程中都要经过：线路建立、数据传输和线路释这 3 个阶段，如图 8-4 所示。从图中可以看出，在“线路建立”和“线路释放”这两个主要阶段中都需要由一方发送请求数据报，然后另一方确认后返回应答数据报，是一个二次握手过程。线路建立好后就进入了“数据传输”阶段，此时接收方也是需要返回确认应答的，否则会在重传定时器超时后重传对应的数据。

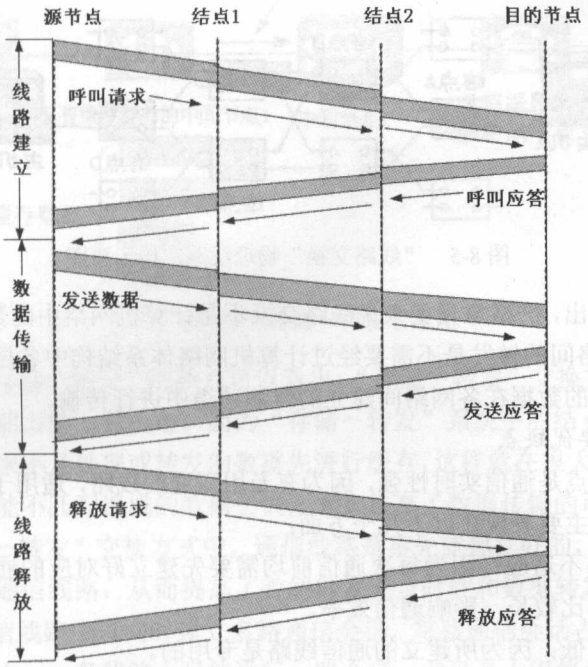


图 8-4　线路交换的 3 个阶段

1. 线路交换的 3 个阶段

（1）线路建立。

在线路交换系统中，源端和目的端之间先要经过呼叫在一级级的网络间建立通信线路，最终建立一条端到端的专用通信线路（或通信通道）。网络结点设备间的呼叫类似“开关”切换方式。

如图 8-5 所示，主机 A 先向与其直接相连的结点 A 提出连接请求，然后结点 A 在通向主机 B 的路径中找到下一个可用的支路。假设结点 A 最终选择经结点 B 的电路，并为此电路分配一个未用的通道，并且告诉结点 B 它还要连接主机 B 节点；结点 B 同样在通向主机 B 的路径中找到下一个可用的支路。假设结点 B 最终选择经结点 C 的电路，也为此电路分配一个未用的通道，并告诉结点 C 它还要连接主机 B 节点；结点 C 按照同样的方法选择一条可用的支路，假设最终选择了结点 D，而主机 B 是直接连接在结点的一个支路上，最终就建立了由主机 A 到达主机 B 的完整通信

线路，用于在主机 A 与主机 B 之间的数据传输。

（2）数据传输。

发送端和接收端之间的通信线路建立好后，即可直接在这条通信线路上进行数据传输了，当然数据的传输也是要经由一级级结点设备的转发的，但不需要经过三层协议封装（直接传输）。

（3）线路释放。

当数据传输完成后，由任一发送端或接收端发出“释放请求”信令。该信令沿通路各结点传送，指示这些结点拆除对应的链路，以释放信道资源。

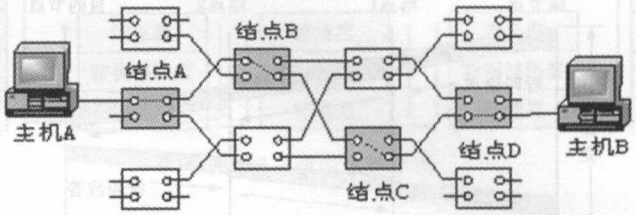


图 8-5 “线路交换”物理线路连接示意图

从以上分析可以看出，严格意义上来说线路交换不是计算机网络中的数据交换方式。因为在线路交换中，数据在各网络间的转发是不需要经过计算机网络体系结构中各层协议的封装和解封装过程的，是直接把要转发的数据在各网络间建立的传输通道中进行传输。

2. 线路交换的主要优缺点

线路交换方式的优点是通信实时性强，因为有专用直通的线路，适用于交互式会话类通信。线路交换的缺点比较多，主要表现在以下几个方面：

- 对突发性通信不适应：因为每次通信前均需要先建立好对应的通信线路，而且线路建立所需要的时间比较长，影响通信效率。
- 线路利用率较低：因为所建立的通信线路是专用的。
- 很难实现拥塞控制：一次通信的所有数据均必须沿着这一条线路进行传输，没有选择余地。路径上的设备也没有缓存功能，所以可能因为不同网络间的传输带宽不匹配而造成数据丢失。
- 维护困难：只要通信双方建立的线路中任何一点出现故障，就必须重新建立新的连接。

正因为线路交换具有以上许多不足，所以线路交换目前很少被采用，主要适用于高负荷的持续通信和实时性要求较强的场合（如会话式通信），不适合突发性通信。

8.2.2 存储－转发原理

“存储－转发”（Store-and-forward）是一种大概念上的三层数据交换机制，总的特点是网络结点设备在接收到数据后，可根据设备 CPU 或对应转发路径下的出接口的当前状态决定是否立即转发。即如果 CPU 或对应的出接口比较繁忙，来不及处理或转发数据，则可先把数据缓存下来，等

待设备 CPU 或者出接口空闲时再进行转发。

支持“存储－转发”三层数据交换技术的结点设备均自带有一定容量的缓存，用于临时存储本地设备接收到了但来不及处理或转发的数据。如图 8-6 所示是一个“报文交换”示例，各中间节点均具有数据缓存能力，中间节点 1 对于来不及处理或发送的数据 1 至数据 4 先缓存在缓存中，等数据 0 发完后，接下来依次发送数据 1 至数据 4。

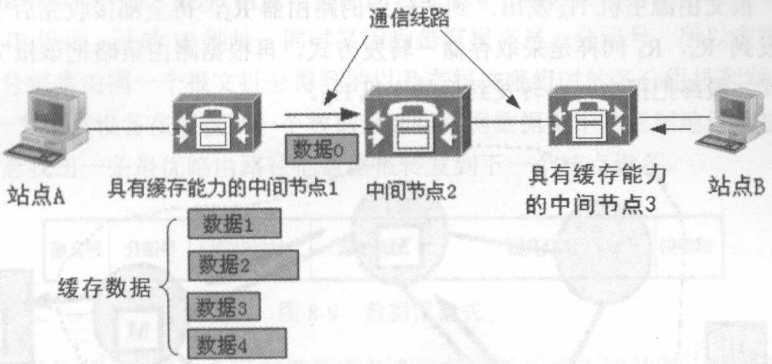


图 8-6 存储－转发示意图

从以上介绍的“存储－转发”的特点可以看出，“存储－转发”交换方式主要是对“线路交换”方式中没有拥塞控制能力的一种改进，因为“存储－转发”系统中的结点设备具备数据缓存能力，可以在接收数据后对来不及处理或转发的数据先进行缓存，这样就在很大程度上避免了因为网络间传输速率或者设备性能不匹配造成的数据丢失，大大提高了数据传输的可靠性和拥塞控制能力。

另外，在“存储－转发”交换方式中，通信线路是事先就建立好的，无需像“线路交换”那样专门为每次通信建立通信线路，从而提高了数据转发的及时性和数据转发效率。“存储－转发”交换机制中，在一条通信线路中可同时建立多路通信会话，每路通信会话占有其中一部分带宽，而在以前的“线路交换”中，一条线路上只能进行一路会话，并且是占用整个线路带宽。

在“存储－转发”这个大的三层交换机制的发展历程中，又先后诞生了两种具体的三层交换技术：“报文交换”和“分组交换”，下面具体介绍。

8.2.3 报文交换原理

“报文交换”（Message Switching）是“存储－转发”技术中最初出现的一种具体的三层交换技术。它除了具备“存储－转发”自身的可先存储再转发的特点外，还有一个最大的特点就是结点设备在进行三层数据转发时，是直接以来自上层的原始数据段作为传输单位，不拆分。很显然，这主要适合当时小容量类型的数据转发情形。

在报文交换方式中，数据的三层转发是通过路由方式的，所以所发送的报文是要经过三层协议封装，带有相应的三层协议地址（如 IP 地址）的，其报文格式如图 8-7 所示。其中的“目的地址”和“源地址”均是指三层地址，可用于根据路由表进行数据转发。

报文结构	报文号	目的地址	源地址	数据	校验和

图 8-7　报文交换方式的报文结构

“报文交换”方式中，数据是采用路由转发方式的，具有路由选路能力，即当源端与目的端存在多条可达路径时，结点设备可根据相应的选路策略为当前数据的转发选择一条最优的转发路径。如图 8-8 所示，报文由源主机 H_A 发出，到达最近的路由器 R_3，待全部接收完后 R_3 再根据路由策略把该报文转发到 R_4，R_4 同样是采取存储－转发方式，再根据路由策略把该报文转发到 R_5，R_5 采取同样的方式，最终把报文一一转发到目的主机 H_B。

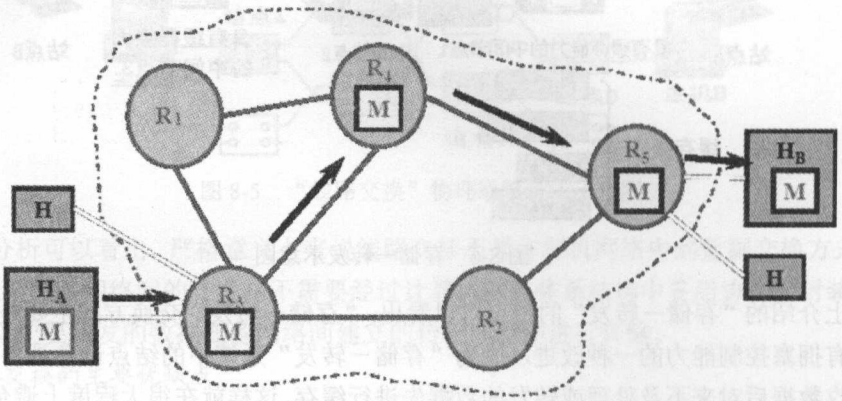


图 8-8　报文交换的路由选路示例

8.2.4　分组交换原理

“分组交换”（Packet Switching，即“包交换”）技术是随着计算机网络应用的发展而新开发的三层交换技术。因为随着多媒体网络应用的出现，用户数据报越来越大，这样一来如果仍采用原来的“报文交换”方式就会经常出现因为数据报大小超过对应数据链路层的 MTU 值而造成大量数据被丢失的现象。于是想到了对大的数据报进行拆分的技术，也就是这里所说的“分组”。

“分组交换”也采用“报文交换”的存储－转发机制，但是规定了传输数据的单位长度，过长的报文被分成较小的单位（分组，Packet），依次发送。现在主要采用这种数据交换方式。在分组交换方式中又有两大技术派系。

一种是采用路由技术（也是目前比较普遍采用的一种交换方式），在要转发的数据报头部加上源节点和目的节点的 IP 地址，然后通过路由技术一级级地把数据转发下去。这种分组交换技术就是通常所说的“数据报”服务，其中的分组称为“数据报”（Datagram）。

另一种是不依靠路由技术，但需要在进行数据分组转发前先在源节点和目的节点间的所有结点设备间建立一条虚拟的通信通道，然后再把数据分组从这个虚拟通道中转发到目的节点。这种分组交换技术就是“虚电路”服务，所建立的虚拟通道称为“虚电路”（Virtual Circuit，VC）。采用虚

电路方式进行分组交换的典型实例为 ADSL 拨号连接。

1. 数据报服务

在“数据报”分组交换方式中，每个分组均被视为一个数据报（Datagram），可单独通过路由路径发送，无需先创建传送的连接（是无连接的服务），但结点设备需要为每个数据报作独立的路由选择。

“数据报”是可以作为独立的数据进行传输的，因为每个数据报自身携带有足够的路由转发信息，其中包括源 IP 地址、目的 IP 地址，同时又因为带有报文号、分组号，所以在接收端可以根据报文号获知哪些分组是由同一个报文拆分得到的以及在报文重组时的各分组排列顺序，其数据格式如图 8-9 所示。一个结点设备在接收到一个数据报后会根据数据报中的三层地址信息和结点设备所配置的路由表信息找出一条最优路由路径把数据报转发到下一个结点设备。

报文号	分组号	目的IP地址	源IP地址	报文分组	校验和

图 8-9 数据报格式

在“数据报”服务中，由同一个报文拆分的各个分组可能采用不同的路由转发路径，如图 8-10 所示就是一个不同数据报沿不同路径传输的示例。从主机 A 发送的到达相同目的节点——主机 B 的两个数据报 P_1、P_2 沿着不同路径进行传输。当然，只是可以这么做，但并不是说数据报分组交换中每个数据报必须走不同的路径。

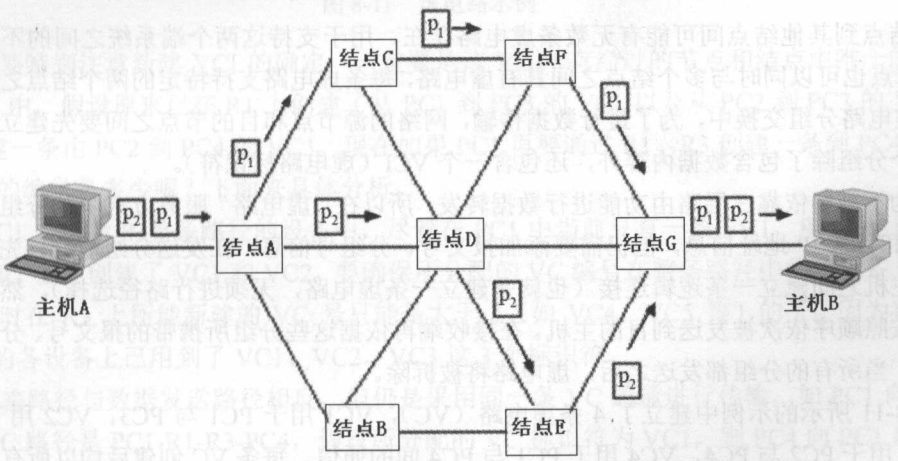


图 8-10 不同数据报沿不同路径进行交换的示例

总体而言，“数据报”服务具有以下几方面的主要特点：

（1）无需构建专门的通信线路，通信效率高。

分组在传送之前不需要预先在源节点与目的节点之间建立“线路连接”，只需根据结点设备（如路

由器、三层交换机）所创建的路由信息为不同的数据报选择相同或不同的传输路径，通信效率较高。

（2）每个分组可以走不同的转发路径，提高了线路的利用率。

由于每个分组都带有路由转发所需的三层地址信息，所以每一个分组都可以独立地选择一条路由路径，同时每一条路径都不是某个分组传输专用的，提高了线路的利用率，也在一定程度上实现了线路的负载分担。

（3）数据有效传输率较低。

由于每一个分组在传输过程中都必须带有目的地址和源地址信息，所以对于一个大的数据报文在拆分成多个分组后，最终在线路上的有效数据传输率比较低。

（4）数据传输延时较大。

因为每一个分组在经过每一个结点时都要根据路由算法选择最佳路由路径，所以传输延迟较大，不适用于长报文、会话式通信。

2. 虚电路服务

前面介绍的“数据报”服务是无连接的服务，在两个节点间通信前无需专门建立通信连接，利用的是无连接的 IP 协议路由功能进行分组转发。此处介绍的“虚电路”服务是面向连接的服务，所以在两个节点间通信前先要建立专门的连接，这类似于前面介绍的“线路交换”技术。但这里的连接建立不是指物理传输线路的建立，而是在原有物理传输线路的基础上构成一条专门的虚电路（Virtual Circuit，VC），并且以一个在所有经过的节点或结点上均唯一的虚电路标识符（Virtual Circuit Identifier，VCI）进行标识。这条 VC 所代表的就是所有经过的节点和结点的串连。

每个结点到其他结点间可能有无数条虚电路存在，用于支持这两个端系统之间的不同数据传输。一个结点也可以同时与多个结点之间具有虚电路，每条虚电路支持特定的两个结点之间的数据传输。在虚电路分组交换中，为了进行数据传输，网络的源节点和目的节点之间要先建立一条逻辑通路。每个分组除了包含数据内容外，还包含一个 VCI（虚电路标识符）。

正因为不需要依靠三层路由功能进行数据转发，所以在“虚电路”服务方式中的分组无需添加源 IP 地址和目的 IP 地址信息，但仍需要添加报文号、分组号信息。在发送分组前，首先在发送主机和目的主机之间建立一条逻辑连接（也就是建立一条虚电路，无须进行路径选择），然后所有的分组都将按照顺序依次被发送到目的主机。在接收端再依据这些分组所携带的报文号、分组号进行报文重组。当所有的分组都发送之后，虚电路将被拆除。

在图 8-11 所示的示例中建立了 4 条虚电路（VC），VC1 用于 PC1 与 PC3，VC2 用于 PC2 与 PC3，VC3 用于 PC2 与 PC4，VC4 用于 PC1 与 PC4 间的通信。每条 VC 创建后均以所有经过的节点和结点唯一的 VCI（虚电路标识符，从 1 开始）自动创建一个 VC 号。

假设 PC1 有一个或多个报文要发送到 PC4 去，那么 PC1 先要发送一个呼叫请求分组到结点 R1，请求建立一条到节点 PC4 的连接。结点 R1 根据它的拓扑连接情况确定到达目的节点 PC4 最佳下一个结点为 R3，与之建立起逻辑连接后把来自 PC1 的呼叫请求分组转发给 R3，结点 R3 最终把呼叫请求分组转发到目的节点 PC4。如果 PC4 准备接收这个连接请求，就发送一个呼叫应答分

组到结点 R3，然后通过与上述相反的路径把呼叫应答分组通过结点 R1 返回到 PC1。这样就在源节点 PC1 和目的节点 PC4 之间建立起了一条逻辑通路 PC1-R1-R3-PC4，然后根据这条路径中所有节点和结点中已分配的 VCI 情况为该条新建的 VC 分配一个对于路径中各节点和结点均唯一的 VCI，这样就完成了一条“虚电路”（VC）的建立过程。

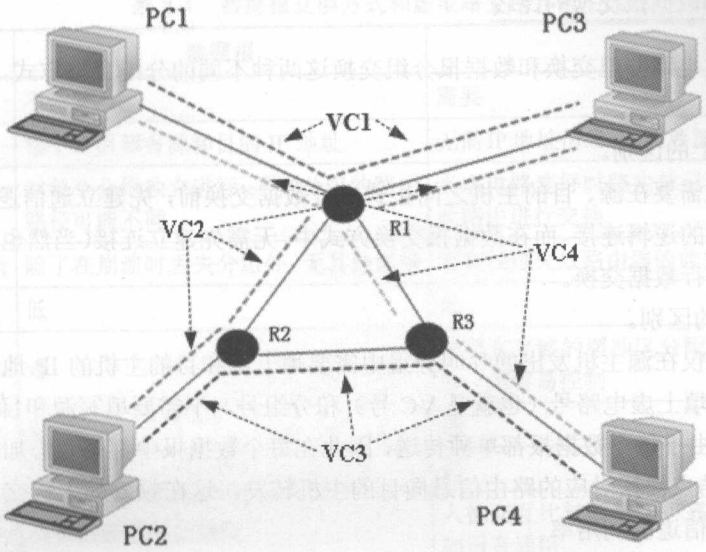


图 8-11 虚电路示例

在这里要特别注意新建 VCI 的确定，一定要是该 VC 所有经过的节点和结点中唯一的 VCI。如在图 8-11 中，假设原来已在 R1 上创建了从 PC1 到 PC3 的 VC1 以及从 PC2 到 PC3 的 VC2，在 R3 上已创建一条由 PC2 到 PC4 的 VC3。现在如果 PC1 再要通过 R1、R3 创建一条到 PC4 的 VC，那这条 VC 的编号是多少呢？下面来具体分析。

在从 PC1 到 PC4 的整条路径的设备中，尽管在 PC1 中当前只有一条 VC1，R3 中也只有一条 VC3，但在 R1 上已创建了 VC1 和 VC2，要确保所分配的 VC 编号在整条路径中的各设备上都是唯一的，则此时在 R3 上所能新建的 VC 号只能是大于 3（如 VC4、VC5 等）的了，因为在新建的 VC 路径中的各设备上已用到了 VC1、VC2、VC3 这 3 个标识符。

返程应答路径与数据发送路径相反，但仍是采用同一条 VC 通道进行传输，如 PC1 向 PC4 发送数据的 VC 路径是 PC1-R1-R3-PC4，假设所分配的 VC 标识符为 VC1，则 PC4 向 PC1 应答的信息传输路径为 PC4-R3-R1-PC1，但仍使用 VC1 这条虚拟通道。

综上所述可以得出，虚电路分组交换方式具有以下几个主要特点：

- 在每次分组交换前，必须在发送方与接收方之间建立一条逻辑连接。
- 一次通信的所有分组都通过这条虚电路（VC）顺序传送，也就相当于有固定的交换路径，所以每个分组均不必带目的 IP 地址、源 IP 地址等信息，但仍必须带有报文号和分组号信息。

- 分组通过虚电路上的每个结点时，结点只需要进行差错检测（进行校验和检测），而不需要进行路径选择。
- 通信子网中每个结点可以和任何结点建立多条虚电路连接，每条物理线路可以建立无数条虚电路连接，每条虚电路支持特定的两个结点之间的数据传输。

8.2.5 虚电路交换和数据报交换的比较

为了正确理解虚电路分组交换和数据报分组交换这两种不同的分组交换方式，本节从各方面对它们进行比较。

（1）传输方式上的区别。

虚电路交换方式需要在源、目的主机之间正式进行数据交换前，先建立通信逻辑连接，通信完成后又需要拆除对应的逻辑连接。而在数据报交换方式中，无需先建立连接（当然也无需释放连接），直接根据路由信息进行数据交换。

（2）数据格式的区别。

虚电路交换方式仅在源主机发出的呼叫分组中需要填上源和目的主机的 IP 地址，在数据传输阶段，各分组都仅需填上虚电路号（也就是 VC 号）和分组号，不需要填写源和目的 IP 地址信息。而数据报交换方式，由于每个数据报都单独传送，因此在每个数据报中都必须添加源和目的主机的 IP 地址，以便网络结点根据对应的路由信息向目的主机转发。这在频繁的人机交互通信应用中不是很适应，也降低了信道的利用率。

（3）转发路径的区别。

虚电路交换方式同一次通信的每个分组都会沿着同样的路径进行转发，也无需进行路径选择。在数据报交换中每个数据报每经过一个网络结点都要进行一次路由选择，而且同一报文所分出的多个报文分组所采用的转发路径也可能不一样。

（4）可靠性方面的区别。

在虚电路交换方式中，由于从源主机发出的所有分组都是通过事先建立好的一条虚电路进行传输，所以能保证分组按发送顺序到达目的主机，而且目的主机每收到一个分组后需要向源主机应答确认，可靠性高。但在数据报交换方式中，当需要把一份长报文分成若干个短的数据报分组时，由于它们都是被独立传送，还可以各自通过不同的路径到达目的主机，因而不能保证这些数据报分组按序列到达目的主机，再加上这种交换方式中目的节点在收到数据报后也不需要发送确认，所以可靠性较低。

（5）适应性方面的区别。

在虚电路交换方式中，当传输途中的某个结点或链路发生故障时，必须重新建立虚电路才能进行通信；而在数据报交换方式中，可以绕开这些故障地区另选其他路由路径快速地把数据传至目的地。因此数据报交换比虚电路交换的适应性更强。

（6）拥塞控制能力方面的区别。

在数据报交换方式中，中间结点可为数据报选择一条流量较小的路由路径，避开流量较高的路

径，因此数据报交换方式既可平衡网络中的信息流量，又可使数据报得以更迅速的传输。而在虚电路交换方式中，一旦虚电路建立后，中间结点是不能根据流量情况来改变分组的传送路径的，所以拥塞控制能力较差。

综上所述，虚电路交换方式和数据报交换方式各有优缺点，综合比较如表 8-1 所示。

表 8-1 数据报交换方式和虚电路交换方式的比较

比较选项	数据报	虚电路
建立连接	不需要	需要
添加地址信息	每个分组都有源和目的 IP 地址	无需 IP 地址信息，但需要加上虚电路标识符（VCI）
路由选择	对每个分组独立进行，不同分组的路由路径可能不同	当虚电路建好时路由就已确定，所有分组都经过此路由进行交换
结点失效的影响	除了在崩溃时丢失分组外，无其他影响	所有经过失效路由器的此虚电路都要被终止
数据传输效率	低	高
拥塞控制	难	如果有足够的缓冲区分配给已经建立的每条虚电路，则容易控制
可靠性	低	高
适应性	强	弱
适用环境	大多数普通数据通信	人机交互比较频繁、可靠性要求高的应用环境，如语音通信

8.3 IPv4 和 IPv6 协议

在 TCP/IP 协议体系结构的“网际互连层”，最重要的协议就是 IP 簇协议。目前主流的 IPv4 协议簇中包括了 3 个协议：IP（Internet Protocol，因特网协议）、ARP（Address Resolution Protocol，地址解析协议）、ICMP（Internet Control Message Protocol，网特网控制消息协议），IPv6 协议簇中包括了 4 个协议：IPv6、ICMPv6、ND（Neighbor Discovery，邻居发现）和 MLD（Multicast Listener Discover，组播侦听器发现）。本节先介绍最主要的网络层协议——IPv4 和 IPv6 协议。

8.3.1 IP 协议基本功能

目前的计算机网络，特别是 TCP/IP 网络，使用最多的是数据报分组交换方式，而 IP 协议是用于将多个分组交换网络连接起来的最典型通信协议。

IP 协议是一个无连接的协议（即在通信前无需建立专门的连接），负责在源地址和目的地址之间传送数据报，然后为了适应不同网络对分组大小的要求，需要对上层传来的报文进行分割，最后调用本地网络协议将数据报传送至下一个网关或目的计算机。经过十几年的发展，目前最新的 IP 协议的版本是 IPv6，不过现在主流使用的仍是 IPv4。

IP 协议的主要功能就是把数据报在互联的网络上传送，通过将数据报在一个个 IP 模块间传送直到目的模块来达到目的。网络中每个主机和网关上都有 IP 模块。数据报在一个个模块间通过路由处理网络地址传送到目的地址。具体来说，IP 协议主要有以下几方面功能：

（1）寻址。

通过第 6 章的学习我们已经知道，在同一以太局域网内部，节点间的寻址可以通过二层 MAC 地址进行，但在不同网络之间不能通过 MAC 地址进行寻址，因为在数据通信中，数据到达不同网络时需要重新进行帧封装，里面的源/目的 MAC 地址是会进行替换的。

在不同网络中只能通过不同网络中统一使用的三层地址进行寻址，就像在一个国家范围内不能通过人的名字来查找一个人，而必须通过身份证来查找一样，因为同名的人可能非常多，而每个人的身份证则是全国唯一的。当然不同协议的网络，三层协议也不一样，如我们常用的 IP 网络中运行的三层协议就是 IP 协议，对应的三层地址就是 IP 地址，而在以前的 Netware IPX/SPX（Internetwork Packet eXchange/Sequences Packet eXchange，网际包交换/顺序包交换）网络中三层协议就是 IPX 地址。

有关 IPv4 和 IPv6 地址方面的知识将在下一章中具体介绍。

（2）数据报的封装。

在 IP 网络中，从传输层到达的数据段都需要经过 IP 协议进行重新封装。因为 IP 协议是无连接的服务，并且采用数据报交换方式，所以封装后形成的是 IP 数据报。IP 封装的目的就是标识此 IP 数据报发送节点和接收节点的 IP 地址及控制信息。有关 IP 数据报的格式将在本章后面具体介绍。

（3）分段与重组。

不同网络上的链路可以传输的最大数据单元（MTU）是不同的，为了使传输的数据报能在不同网络中传输，当一些尺寸较大的数据报到达某个 MTU 值比较小的网络链路上传输时就可能需要对原来的数据报进行拆分，形成一个个小的分段，然后再把这些分段依次传输出去，即 IP 协议的“分段”（或分组）功能。在接收节点又要把这些被拆分的分段重新组合起来，还原成原来的大的数据报，这就是 IP 协议的分段“重组”功能。

8.3.2 IPv4 的不足

IPv4 自 1981 年发布 RFC791 后就没有什么更改。虽然 IPv4 被无数实践证明是可靠的、易于实现和可交互的，并且经受了各种考验，但时至今天随着 IP 网络应用的普及和高速发展，当初设计的 IPv4 仍面临以下这些无法解决的实际问题：

（1）IPv4 地址空间面临枯竭。

我们知道，IPv4 地址分为私网地址和公网地址两大类，私网地址就是仅供局域网内部使用的地址，不同企业都可以重复使用，而公网地址则是全球唯一的，是需要注册、购买的，一个地址只能在一个地方使用。由此可见，公网 IP 地址可以说是用一个少一个。但 IPv4 地址仅 32 位，总体可提供的地址数都比较有限，更不用说公网地址了。这是 IPv4 最终走向死亡的最大因素。

公网 IPv4 地址已经变得相当缺乏，时至今日几乎到了全球都无 IP 地址可分了，这会大大影响计算机网络今后的发展，也严重不适应当前移动互联网、物联网等新兴互联网技术的发展和普及。尽管我们一直在采用像 NAT（Network Address Translation，网络地址转换）、VLSM（Variable Length Subnet Mask，可变长子网掩码）、CIDR（Classless Inter-Domain Routing，无类域间路由）这类可以更加充分使用现有公网 IP 地址的技术，但仍然很难满足从企业到个人的新兴互联网应用需求，于是采用新的 IPv6 协议的呼声在全球都是日益高涨。

（2）骨干路由器维护的路由表项数量过大。

由于 IPv4 发展初期分配规划的问题，造成许多 IPv4 地址块分配不连续，不能有效聚合路由。针对这一问题，虽然通过采用 CIDR 比较有效地抑制了全球 IPv4 BGP 路由表的线性增长，但目前全球 IPv4 BGP 路由表仍在不断增长，经过 CIDR 聚合以后的 BGP 路由表项有的也已高达几十万条。日益庞大的路由表耗用内存较多，对设备成本和转发效率都有一定的影响，这一问题促使设备制造商不断升级其路由器产品，提高其路由寻址和转发的性能。

（3）地址结构不合理。

IPv4 地址的 A、B、C、D 和 E 这 5 种分类结构本身就存在严重缺陷，不利于地址资源的充分利用。如果一个组织分配了 A 类地址，大部分的地址空间就被浪费了；如果一个组织分配了 C 类地址，地址空间又严重不足，而且 D 类和 E 类地址都无法利用。虽然可以通过 VLSM 和 CIDR 技术来弥补，但这样同样会使路由表和路由策略变得十分复杂，大大影响路由效率。

（4）配置复杂。

目前大多数 IPv4 的实现方案必须手动配置或者通过控制状态的地址配置协议进行配置，例如通过动态主机配置协议（DHCP）进行配置。随着使用 IP 的计算机和设备越来越多，越来越需要更简单更自动化的地址配置和其他不依赖于 DHCP 基础结构的管理配置设置。IPv6 协议就有这种自动配置功能。

（5）不支持端到端安全。

在 IPv4 报文格式中像 IPSec 安全功能是可选项，在大多数基于 IPv4 协议的网络应用的安全功能就需要依靠其他安全协议或功能模块来实现，所以 IPv4 协议自身的安全性比较差。在 IPv6 报文格式中就自带有像 IPSec 这样的安全功能，所以基于 IPv6 的应用中基本上不需要配置额外的安全功能。如在 IPv4 网络中的 RIP、OSPFv2 和 IS-IS 路由协议中邻居建立的安全认证就需要另外配置，而在 IPv6 网络中，RIPng、OSPFv3 等路由协议的邻居建立的安全保护功能不需要另外配置，可直接使用 IPv6 协议自身的安全功能。

8.3.3 IPv6 的主要优势

为了解决以上 IPv4 的这些不足，Internet 工程任务组（IETF）已经开发了一组协议和标准，那就是 IPv6 协议簇，其中最主要的协议当然就是 IPv6 了。IPv6 与 IPv4 协议一样，也是一种无连接的、不可靠的数据报协议，主要用于在主机之间寻址和路由数据报。

新的 IPv6 协议相对目前正在使用的 IPv4 协议来说主要具有以下几方面的新特性：

（1）更大的地址空间。

IPv6 的地址结构中有 128 位（16 个字节），比起 IPv4 的 32 位（4 个字节）来说扩展了 4 倍，可以极大地满足未来新兴互联网技术和应用发展的地址需求。初步估算，全球每个人都可以分到差不多 10 个专用 IP 地址，到那时我们每个人都可以在互联网上有唯一的身份了。

（2）有效和分级的寻址及路由基础结构。

IPv6 地址长度为 128 位，可提供远大于 IPv4 的地址空间和网络前缀，因此可以方便地进行网络的层次化部署。同一组织机构在其网络中可以只使用一个前缀。对于 ISP，可获得更大的地址空间。这样 ISP 可以把所有客户聚合形成一个前缀并发布出去。分层聚合使全局路由表项数量很少，转发效率更高。另外，由于地址空间巨大，同一客户使用多个 ISP 接入时可以同时使用不同的前缀，这样不会对全局路由表的聚合造成影响。

（3）支持无状态和控制状态的地址配置。

为了简化主机配置，IPv6 既支持控制状态的地址配置，例如 DHCP 服务器存在时的地址配置，又支持无状态的地址配置，即 DHCP 服务器不存在时的地址配置。使用无状态的地址配置，链接上的主机将使用用于该链接的 IPv6 地址（本地链接地址）和由本地路由器广告的前缀导出的地址自动配置它们自己。即使没有路由器，同一个链接上的主机也可以自动用本地链接地址配置自己，并在没有手动配置的情况下通信。

（4）对服务质量（QoS）的更好支持。

IPv6 报头中的新字段定义了如何处理和识别通信。通过在 IPv6 报头中使用“流标签”字段，通信标识允许路由器识别属于流的数据报并为之提供特殊处理。因为在 IPv6 报头中标识通信，所以即使在数据报有效负载使用 IPSec 加密时也很容易实现对 QoS 的支持。

（5）提供用于邻近节点交互的新协议。

IPv6 的 ND（邻居发现）协议是一个用于 IPv6 的 Internet 控制消息协议，用于管理邻居节点（即同一链接上的节点）的交互。ND 协议使用有效的多播和单播消息代替原来 IPv4 协议中的 ARP 和 ICMPv4，并提供附加的功能。

（6）可扩展性。

IPv4 中虽然也支持 IPSec，但只是通过选项支持，实际部署中多数节点都不支持。但在 IPv6 中，IPSec 是 IPv6 协议基本定义中的一部分，部署的任何节点都必须支持。因此，在 IPv6 中支持端到端安全要容易得多。IPv6 中支持为 IP 定义的安全目标：保密性（只有预期接收者能读数据）、完整性（数据在传输过程中没有被篡改）、验证性（发送数据的实体和所宣称的实体完全一致）。

（7）支持移动特性。

IPv6 协议规定必须支持移动特性，任何 IPv6 节点都可以使用移动 IP 功能。与移动 IPv4 相比，移动 IPv6 使用邻居发现功能可直接实现外地网络的发现并得到转交地址，而不必使用外地代理。同时，利用路由扩展头和目的地址扩展头移动节点和对等节点之间可以直接通信，解决了移动 IPv4 的三角路由、源地址过滤问题，移动通信处理效率更高且对应用层透明。

8.3.4 IPv4 数据报头部格式

IPv4 协议是无连接的网络层协议，目前基本上采用前面介绍的分组交换方式，里面封装的是一个个携带有源/目的 IP 地址信息的数据报。发送端的网络层在收到来自它的上一层——传输层发来的数据段时需要进行 IP 协议封装，包括源/目的 IP 地址，以便可以通过 IP 寻址把数据报通过网络一级级路由传输到目的网络中。

与在第 7 章中介绍的数据帧格式中包括“帧头”和“数据”部分类似，一个 IP 数据报也包括“报头”和“数据”这两个组成部分，如图 8-12 所示。其中“数据”部分就是来自上层的数据，而“报头”部分是为了正确传输数据报而增加的网络层 IPv4 协议信息。

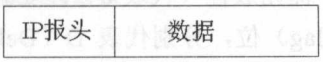


图 8-12 IPv4 数据报基本结构

【说明】在网络层中有许多协议，而且这些协议在网络层内部又有层次高低之分，其中的 IP 协议属于中等层次，比 IP 协议层次高的网络层协议有 ICMP、IGMP 等协议，比 IP 协议层次低的网络层协议有 ARP 协议。凡是比 IP 协议层次更高的协议的报文都需要经过 IP 协议封装，而比 IP 协议层次更低的协议的报文又不需要经过 IP 协议封装，如 ARP 协议报文就无需经过 IP 协议封装。所以前面说的“上层”是既包括同时位于网络层但比 IP 协议层次更高的 ICMP、IGMP 等协议，又包括更高层的传输层协议。

1. IPv4 数据报头部格式

IPv4 数据报头部格式如图 8-13 所示。

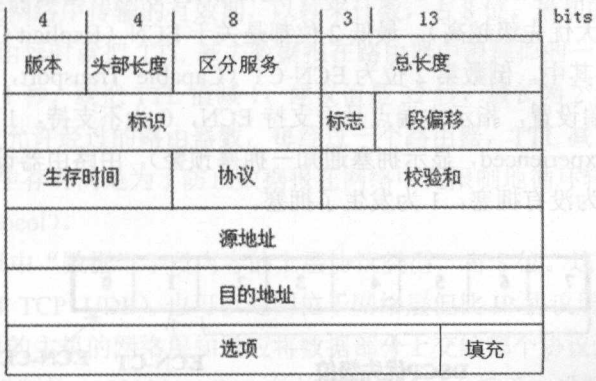


图 8-13 IPv4 数据报头部格式

下面对各个字段进行说明。

（1）版本（Version）。

指定 IP 数据报中使用的 IP 协议版本，占 4 位。此处是 IPv4 版本，值为 4（0100）。

（2）头部长度（Header Length）。

指示 IP 数据报头部的总长度，以 4 个字节为单位，占 4 位。当报头中没有“选项”字段时，报头的总长度为 5，即 20 个字节（此时报头长度值为 0101）。当 IP 数据报头部长度为 1111 时（即十进制的 15），头部长度就达到了 60 个字节，此时报头部分肯定包括一些可选项。但报头长度必须是 32 位（4 个字节）的整数倍（因为该字节是以 4 个字节为单位进行计数），否则需要在“选项”字段中的“填充”（PAD）子字段中加 0 凑齐。

（3）区分服务（Differentiated Services）。

在最初的 IP 数据报中，这个字段为“优先级和服务类型”，也称“服务类型”（ToS）字段，用于表示 IP 数据报的优先级和服务类型，占 8 位，具体结构如图 8-14 所示。其中，最高 1 位未用（固定为 0），随后 3 位为 IP 优先级位（代表 0～7 共 8 个优先级值，默认为 0，值越大优先级越高），最低 4 位是标志（Flag）位，分别代表 D（Delay，延迟）、T（Throughput，吞吐量）、R（Reliability，可用性）和 C（Cost，开销），置 1 时表示要优先考虑对应方面，以获得相应的服务级别。

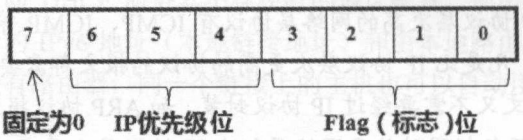


图 8-14　ToS 字段格式

1998 年 IETF（国际互联网工程任务组）在 RFC2474 中把 IP 数据报中的 ToS 字段改名为“区分服务”（DS）字段，同样为 8 位，但是它与 ToS 字段结构不一样，如图 8-15 所示。其中，最高 6 位构成 DSCP（Differentiated Services Code Point，区分服务码点），定义了 0～63 共 64 个 DSCP 优先级（默认为 0，值越大优先级越高），最低 2 位都是关于 ECN（Explicit Congestion Notification，显示拥塞通知）的位。其中，倒数第 2 位为 ECN-CT（Capable Transport，显示拥塞通知—能力传输），由发送数据的源端设置，指示本端点是否支持 ECN，0 为不支持，1 为支持；倒数第 1 位为 ECN-CE（Congestion Experienced，显示拥塞通知—拥塞预警），由路由器设置，用来向终端站点指示是否发生了拥塞，0 为没有拥塞，1 为发生了拥塞。

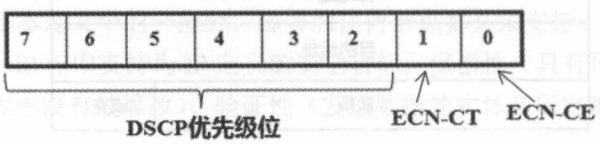


图 8-15　DS 字段格式

（4）总长度（Total Length）。

标识整个 IP 数据报的总长度（包括报头和数据部分），以字节为单位，占 16 位。由此可以得出，IPv4 数据报的最大长度为 2^{16}-1（65535）字节（即 64KB）。之所以要减 1，是因为 2^{16} 中包括

0 这个值。

【说明】在网络层下面的每一种数据链路层都有其自己的帧格式，其中包括表示“数据”字段的最大长度，这称为最大传送单元（Maximum Transfer Unit，MTU）。当一个数据报封装成链路层的帧时，此数据报的总长度（包括“头部”和“数据”两部分）一定不能超过下面的数据链路层的 MTU 值，否则数据报要进行拆分。具体将在下节介绍 IP 数据报封装时介绍。

（5）标识（Identification）。

标识 IP 数据报的编号，16 位。每发送一个 IP 数据报其编号增加 1，主要用于数据分组的组装。当一个数据报由于其长度超过下面数据链路层的 MTU 值而必须拆分时，这个标识字段的值就被复制到所有的数据报分组的“标识”字段中。这样到了接收端，带有相同“标识”字段的值的各数据报分组最后能正确地重装成为原来的数据报。

（6）标志（Flags）。

指示该 IP 数据报后面是否还有分段，是分段标志，占 3 位。目前只有最低 2 位有意义，最高 1 位保留，中间 1 位为 DF（Don't Fragment，不分段）位，置 1 表示不进行分段，置 0 表示进行分段；最低 1 位为 MF（More Fragment，更多分段）位，置 1 表示后面还有分段，置 0 表示后面没有分段了，本分段为某 IP 数据报的最后一个分段。

（7）段偏移（Fragment Offset）。

以 8 个字节为单位指出本分段的“数据”字段起始字节相对整个 IP 数据报“数据”字段的偏移单位（上层协议头必须全部在第一个分段中），占 13 位。即每个分段中“数据”字段的长度一定是 8 个字节（64 位）的整数倍（不够时填充），每偏移 8 个字节值增加 1；无分段，则该字段值为 0。

（8）生存时间（Time To Live，TTL）。

标识 IP 数据报在网络中传输的有效期，以秒来计数，占 8 位。最初的设计是以秒作为 TTL 的单位，每经过一个路由器时就把 TTL 减去数据报在路由器中消耗掉的一段时间。若数据报在路由器中消耗的时间小于 1 秒，就把 TTL 值减 1，建议值是 32 秒，最长为 2^8-1=255 秒。现在通常认为这个 TTL 是指数据报允许经过的路由器数，每经过一个路由器，TTL 减 1，当 TTL 值为 0 时就丢弃这个数据报。设定生存时间是为了防止数据报在网络中无限制地循环转发。

（9）协议（Protocol）。

标识此 IP 数据报中“数据”字段内容的上层协议类型，占 8 位。这里所说的“上层协议”既可以是传输层协议（如 TCP、UDP），也可以是同位于网络层但比 IP 协议层次更高的协议（如 ICMP、IGMP 等），以便使目的主机的网络层知道应将数据部分上交给哪个协议进行处理。如 ICMP 的协议号是 0x1，等于二进制的 00000001，TCP 的协议号是 0x6，等于二进制的 00000110，UDP 的协议号是 0x17，等于二进制的 00010001。

（10）校验和（Checksum）。

检验 IP 数据报的“报头”部分（不包括“数据”部分）在传输到接收端后是否发生了变化，占 16 位。这是因为数据报每经过一个路由器，路由器都要通过 CRC 重新计算一下报头校验和（一

些字段，如生存时间、标志、段偏移等都可能发生变化），不检验“数据”部分可减少计算的工作量。



利用“校验和”字段检验报头部分数据正确性的基本原理是先在发送端“校验和”字段中填上一个特定的值，然后在接收端把包括“校验和”字段在内的报头部分进行二进制反码求和，再取反，如果结果为 0 则表示报头部分在传输过程中没有发生变化，否则表示在传输过程中出现了差错。从以上可以看出，这里最关键的是在发送端计算出这个“校验和”值。它的计算步骤如下：

①在发送端，最初把 IP 数据报报头中的“校验和”字段置 0。然后把头部看成以 16 位（2 个字节）为单位的数字组成，对每 16 位的二进制反码进行求和。如报头长度不是 16 比特的倍数，则用 0 比特填充到 16 比特的倍数。此时“校验和”字段值为 0，可以不计，因为 0 的反码仍为 0。以上得到的结果就是我们要求的“校验和”字段值，系统自动填入到 IP 数据报报头的“校验和”字段中。

②在接收端，同样按照以 16 位为单位对 IP 数据报报头部分进行二进制反码求和，再取反，如果结果为 0 则表示报头部分在传输过程中没有发生改变，否则表示发生了差错。

注意，此时因为“校验和”字段已不再是 0 了，而是等于除了“校验和”字段外其他字段的反码之和。对“校验和”字段值取反求和后，再与其他字段的反码之和（相当于原来“校验和”字段的值）相加，结果肯定是全为 1，因为这两个值互为反码；再取反后，结果肯定为 0。这就是校验和的基本原理。

举个简单的例子，假设有 3 个数（为了简便，在此均用 4 位表示）：2（0010）、3（0011）、C（代表“校验和”字段值），其中 C 中是 2 和 3 的反码之和，得到 9（1001）。现在假设把这 3 个数传送到接收端。在接收端也要对这 3 个数进行反码求和。因为 2 和 3 这两个的反码之和我们在计算 C 时就已经得出了，就是 9（1001），现在只需对 C（“校验和”字段值）进行求反，得到 6（0110）。把 1001 与 0110 相加，得到 15（1111），然后再取反就得到了 0。这就是这 3 个数在传输过程没有出现差错的情况下得到的，这就是“校验和”的校验原理。

（11）源地址/目的地址（Source Address/Destination Address）。

分别表示该 IP 数据报发送端和接收端的 IPv4 地址，各占 32 位。在整个数据报传送过程中，无论经过什么路由，无论如何分段，这两个字段一直保持不变。

（12）选项（Options）。

提供各种扩展选项，用来支持排错、测量、安全等措施，1～40 个字节，且可变长。作为选项，用户可以使用也可以不使用它们。但作为 IPv4 协议的组成部分，所有实现 IPv4 协议的设备都必须能处理 IPv4 选项。在使用选项的过程中，如果造成 IPv4 数据报的报头不是 32 位的整数倍，这时需要使用后面的“填充”字段凑齐。如果恰好为整数倍，则不需要“填充”字段。



从图 8-13 可以看出，在 IPv4 数据报中关于地址信息方面只有源和目的 IP 地址，并没有给出对应的子网掩码。而我们知道，IP 数据报的路由转发依据是 IP 数据报中的目的 IP 地址，但 IP 数据报中仅有 IP 地址没有子网掩码，又该如何确定所要使用的路由表项呢？其实这就是路由表选择中最重要的一个“最长匹配”原则的由来了。也就是当设备上有多条路由表项与要被转发的 IP 数据报中的目的 IP 地址匹配时，选择其中掩码长度最长的那个路由表项为该 IP 数据报进行转发。这就增加了 IP 数据报转发的灵活性，也大大提高了 IP 数据报转发的可靠性，减少了设备中所需的路由表项数量，否则每个目的 IP 网段都必须有一条明确的路由表项。

2. IP 数据报报头示例

如图 8-16 所示是在科来分析仪上通过抓包得出的 IP 数据报各字段的值示例，这也可以全面体现图 8-13 所示的 IP 数据报头部格式。

```
⊟ Internet Protocol, Src: 192.168.0.10 (192.168.0.10), Dst: 192.168.0.12 (192.168.0.12)
    Version: 4
    Header length: 20 bytes
  ⊟ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
      0000 00.. = Differentiated Services Codepoint: Default (0x00)
      .... ..0. = ECN-Capable Transport (ECT): 0
      .... ...0 = ECN-CE: 0
    Total Length: 60
    Identification: 0x1b53 (6995)
  ⊟ Flags: 0x02 (Don't Fragment)
      0... .... = Reserved bit: Not set
      .1.. .... = Don't fragment: Set
      ..0. .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 128
    Protocol: ICMP (1)
  ⊟ Header checksum: 0x5e07 [correct]
      [Good: True]
      [Bad: False]
    Source: 192.168.0.10 (192.168.0.10)
    Destination: 192.168.0.12 (192.168.0.12)
```

图 8-16　IP 数据报头部格式示例

下面是从中发现的各字段值。

- 版本（Version）：4，表示当前网络中为 IPv4。
- 头部长度（Header Length）：20 个字节。
- 区分服务字段（Differentiated Services Field）：00000000，表示当前 IP 数据报中没有使用服务类型字段，采用默认的“尽力传输”优先级别。
- 总长度（Total Lenth）：60，表示该数据报总长度为 60 个字节。
- 标识（Identification）：表示该数据报的标识为 0x1b53（十六进制）。
- 分段标志（Flags）：0x02，第 2 位为 1，表示该数据报不能被分段，最低位为 0，表示后面没有分段。
- 分段偏移（Fragment offset）：0，表示没有被分段。
- 生存时间（Time to Live）：128，表示该数据报最多可以经过 128 个路由。

- 上层协议（Protocol）：ICMP，值为 0x1。
- 校验和（Header checksum）：0x5e07，correct 表示校验正确。
- 源 IP 地址（Source）：192.168.0.10，表示发送该数据报的源节点 IP 地址为 192.168.0.10。
- 目标 IP 地址（Destination）：192.168.0.12，表示该数据报发送的目的节点 IP 地址为 192.168.0.12。

8.3.5 IPv6 数据报头部格式

RFC 2460 定义了 IPv6 数据报格式。总体结构上，IPv6 数据报格式与 IPv4 数据报格式是一样的，也是由“IP 报头”和“数据”（在 IPv6 中称为“有效载荷”）这两部分组成，但在 IPv6 数据报的“数据”部分还可以包括 0 个或多个 IPv6 扩展报头（Extension Headers），如图 8-17 所示。“IP 报头”部分固定为 40 个字节长度，而“有效载荷”部分最长不得超过 65535 个字节（64KB）。

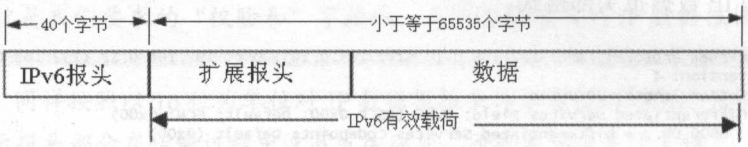


图 8-17　IPv6 数据报结构

IPv6 和 IPv4 之间最显著的区别为：IP 地址的长度从 32 位增加到 128 位。

通过裁减 IPv4 报头中的某些字段或把一些字段移入到扩展报头中，IPv6 基本报头的最大总长度大大减小了。IPv6 使用固定长度的基本报头，从而简化了转发设备对 IPv6 报文的处理，提高了转发效率。尽管 IPv6 地址长度是 IPv4 地址长度的 4 倍，但 IPv6 基本报头的长度只有 40 个字节，是固定的 IPv4 报文头长度（不包括“选项”字段）的两倍。

IPv6 报头格式如图 8-18 所示。

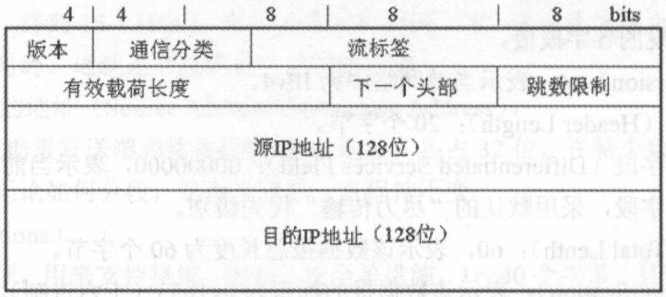


图 8-18　IPv6 报头格式

（1）版本（Version）。

表示 IP 数据报使用的是 IPv6 协议封装，占 4 位，对应值为 0x6（0110）。

（2）通信分类（Traffic Class）。

标识对应 IPv6 数据报的通信流类别或者优先级别，占 8 位，类似于 IPv4 数据报中的 DS（区分服务）字段。

- 前 6 位构成 DSCP（Differentiated Services Code Point，区分服务代码点）优先级，默认为 0，表示尽力传输服务。
- 倒数第 2 位：ECT-Capable Transport（显示拥塞通知－能力传输），由发送数据的源端设置，用来指示传输协议的端点是否支持 ECN，0 为不支持，1 为支持。
- 倒数第 1 位：ECN-CE（Congestion Experienced，显示拥塞通知－拥塞预警），由路由器设置，用来向终端站点指示是否发生了拥塞，0 为没有拥塞，1 为发生了拥塞。

（3）流标签（Flow Label）。

标记报文的数据流类型，以便在网络层区分不同的报文，占 20 位，默认为全 0。

该字段由源节点分配，通过“流标签”“源地址”、“目的地址”三元组方式就可以唯一标识一条通信流，而不用像 IPv4 那样需要使用五元组方式（源地址、目的地址、源端口、目的端口和传输层协议号）。这样有两大好处：一是“流标签”可以和任意的流关联，需要标识不同类型的流时无需对流标签做改动；二是流标签在 IPv6 基本报头中，使用 IPSec 时此域对转发路由器可见，因此转发路由器在使用 IPv6 报文 IPSec 的情况下仍然可以通过三元组（流标签、源地址、目的地址）针对特定的流进行 QoS（质量服务）处理。

（4）有效载荷长度（Payload Length）。

以字节为单位标识 IPv6 数据报中“有效载荷部分”（包括所有“扩展报头”部分）的总长度，也就是除 IPv6 基本报头以外其他部分的总长度，占 16 位。

（5）下一个头部（Next Header）。

标识当前报头（或扩展报头）的下一个头部的类型，占 8 位。每种“扩展报头”都有其对应的值，具体将在下节介绍。

（6）跳数限制（Hop Limit）。

与 IPv4 报文中的 TTL 字段类似，指定了报文可以有效转发的次数，占 8 位。最大值为 255，报文每经过一个路由器节点，跳数值就减 1，当此字段值减到 0 时直接丢弃该报文。



同一路由器上所连接的多个网段之间的跳数是为 0 的，也就是在同一路由器上所连接的多个网络是直接相通的，不用配置路由。这也是像网关、路由器这类设备通过它们上面的多个端口就可以连接多个不同网段的原因。但是不同网关、路由器上连接的不同网段是必须要配置路由的。

另外，之所以称之为“网关”，那是因为在这类设备上仅支持直接连接的多个网段（每个网络连接在一个端口上）间通过它本身即可实现的互通，不提供到达非直连网络（也就是在其他三层设备上连接的网络）的路由功能。但要注意的是，我们在计算机等设备上配置的网关并不是指与它所直接连接交换机端口的 IP 地址，而是指三层设备上连接该计算机所在网段的那个端口的 IP 地址。

“网关”IP 地址必须是位于三层设备（如网关、路由器、三层交换机）端口上的，而且是必须位于网络边缘（即该设备的其他端口必须连接的是不同的网段）的端口上。

（7）源 IP 地址（Source IP Address）。

标识发送该 IPv6 报文源节点的 IPv6 地址，占 128 位。

（8）目的 IP 地址（Destination IP Address）。

标识 IPv6 报文接收节点的 IPv6 地址，占 128 位。

对比图 8-13 中的 IPv4 数据报格式可以看出，IPv6 去除了 IPv4 报头中的“头部长度”“标识”“标志”“分段偏移”“校验和”“选项”和“填充”这么多字段，却只增加了“流标签”这一个字段，因此 IPv6 报头处理比 IPv4 大大简化，提高了处理效率。另外，IPv6 为了更好地支持各种选项处理，提出了扩展头的概念，新增选项时不必修改现有结构就能做到，理论上可以无限扩展，体现了优异的灵活性。

8.3.6 IPv6 扩展报头

在图 8-18 中可以看到，IPv6 报文中是可以携带可选的“IPv6 扩展报头”的。IPv6 扩展报头是跟在 IPv6 基本报头后面的可选报头。由于在 IPv4 的报头中包含了几乎所有的可选项，因此每个中间路由器都必须检查这些选项是否存在。在 IPv6 中，相关选项被统一移到了扩展报头中，这样中间路由器不必处理每一个可能出现的选项（仅有“逐跳选项”报头是必须要处理的），提高了路由器处理数据报文的速度，也提高了其转发性能。

1. 主要 IPv6 扩展报头类型

IPv6 扩展报头是附加在 IPv6 报头“目的 IP 地址”字段后面的，可以有 0 个或者多个扩展报头。扩展报头主要包括：TCP、UDP、ICMP 等上层协议头，以及“逐跳选项报头”“目的选项报头”“路由报头”“分段报头”“认证报头”和“封装安全净载报头”等。

（1）逐跳选项报头（Hop-by-Hop Options Header）。

本扩展报头类型值为 0（在 IPv6 报头 Next Header（下一个头部）字段中定义，下同），用于为通向目标路径上的每次跳转指定所需发送的参数。此扩展报头必须被转发路径上的所有节点（包括源节点和目的节点）处理。目前在路由告警（RSVP 和 MLDv1）和 Jumbo 帧处理中使用了逐跳选项报头，因为路由告警需要通知到转发路径中的所有节点，而 Jumbo 帧是长度超过 65535 个字节的报文，传输这种报文需要转发路径中的所有节点都能正常处理。

（2）目的选项报头（Destination Options Header）。

本扩展报头类型值为 60，只可能出现在两个位置：①“路由报头”前，这时此选项报头被目的节点和路由报头中指定的节点处理；②“上层报头”前，此时只能被目的节点处理。

移动 IPv6 中使用了“目的选项报头”，称为“家乡地址选项”。家乡地址选项由目的选项报头携带，用于移动节点离开家乡后通知接收节点此移动节点对应的家乡地址。接收节点收到带有家乡地址选项的报文后，会把家乡地址选项中的源地址（移动节点的家乡地址）和报文中的源地址（移动节点的转交地址）交换，这样上层协议始终认为是在和移动节点的家乡地址通信，实现了移动漫游功能。

【说明】“逐跳选项报头”和“目的选项报头”有相同的格式，包括以下 3 个部分：

- 下一个头部（Next Header）：占 8 位，指定下一个扩展头的协议号，标识紧随着本扩展头之后的下一个扩展头的协议类型。
- 扩展头部长度（Hdr Ext Len）：占 8 位，以 8 个字节为单位的头部长度，但不包括本扩展头部的 8 个字节，即对于一个 8 个字节的逐跳选项报头或者目的选项报头来说，其扩展头部长度字段的值为 0。
- 选项（Option）：长度可变，但必须使整个扩展报头的长度为 8 个字节的整数倍，标识要发送的参数信息。

（3）路由报头（Routing Header）：本扩展报头类型值为 43，用于源路由选项和移动 IPv6。

（4）分段报头（Fragment Header）：本扩展报头类型值为 44，用于标识数据报的分段，在 IPv4 报头中也有对应的字段。当源节点发送的报文超过传输链路 MTU（源和目的之间传输路径的 MTU）时，需要对报文进行分段时使用。

（5）认证报头（Authentication Header）：本扩展报头类型值为 51，用于 IPSec，提供报文验证、完整性检查。

（6）封装安全有效载荷报头（Encapsulating Security Payload Header）：本扩展报头类型值为 50，用于 IPSec，提供报文验证、完整性检查和加密。

（7）上层报头：用来标识数据报中上层协议的类型，如 TCP、UDP、ICMP 等。

2. 扩展报头使用注意事项

- 如果 IPv6 报文有“逐跳选项”扩展报头，则必须是第一个扩展报头，其他扩展报头位置不限。
- 除了“目的选项”扩展报头可能出现一次或两次（一次在路由报头之前，另一次在上层协议数据报文之前），其余扩展报头只能出现一次。
- 前一个报头的 Next Header 字段指明下一个扩展报头的类型，形成链状的报头列表。
- 每个扩展报头长度不限，但都是 8 个字节的整数倍。

8.3.7 IPv4 数据报的封装与解封装

在 IPv4 网络中，发送端接收到来自传输层的数据段经过网络层时需要进行网络层 IP 协议头部封装（把整个传输层数据段当作 IP 数据报的“数据”字段部分），以便数据报在网络中传输时能根据报文中的“目的 IP 地址”选择适宜的路由表项一级级地向目的网络转发。这就是 IP 数据报的封装过程。

当然封装了 IP 协议头部的数据报还需要继续向下面的数据链路层传输，这时又得在数据链路层进行第 6 章中介绍的“帧封装”（加装对应的链路层协议头部和尾部，把整个网络层 IP 数据报当作“数据”字段部分），数据帧再要向下面的物理层传输，把帧转换成一个个比特位在传输介质中传输。整个流程如图 8-19 所示。

数据从传输介质进入到了对端设备时就要再从物理层接收这一个个比特流，如果数据是来自发

送端的其他高层，则还继续向上层传输。首先当然是要上传到数据链路层，这时要进行“帧组装”，因为物理层的传输单位是一个个比特位，而数据链路层的传输单位是帧。到了数据链路层后，如果在发送端，该数据是来自网络层或者更高层次，则还要继续向上传，首先当然就是向网络层传，这又涉及到“帧解封装”，就是要去掉帧中原来携带的帧头和帧尾，还原出在发送端网络层原始的 IP 数据报格式。有关“帧组装”和“帧解封装”的具体流程参见第 6 章。

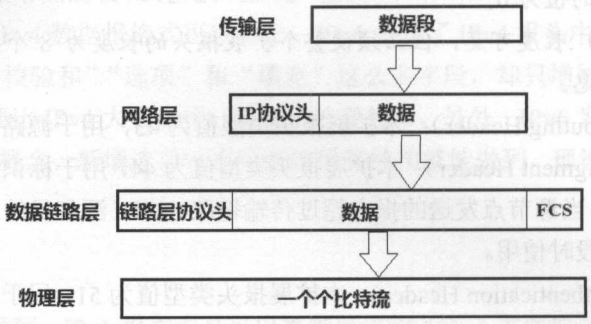


图 8-19　IP 数据报封装流程

此处所说的 IP 数据报“解封装”是指到达网络层的数据报还需要继续向上面的传输层传输时，要去掉 IP 数据报中的 IP 协议头，还原在发送端传输层发送的原始数据段。在发送端整个数据的解封装流程就是图 8-19 的反过程。

下面通过一个实例来说明 IP 数据报的封装和解封装流程。如图 8-20 所示，源主机要发送一个 IP 数据报经过两个路由器连接的 3 个网络（网络 1～3）到达目的主机。

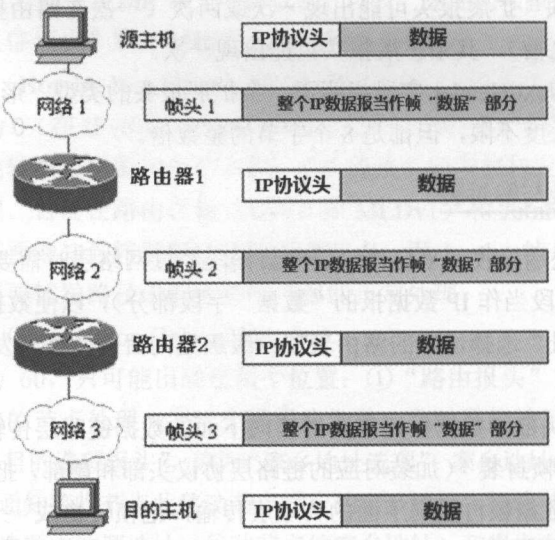


图 8-20　IP 数据报经过多个网络时的封装和解封装流程

因为一般的路由器工作的最高层次就是网络层，所以在各路由器上所进行的数据处理只有帧和 IP 数据报的处理，具体流程如下：

（1）当源主机要向目的主机发起通信时，首先会在该主机的网络层形成一个 IP 数据报，然后向下面的数据链路层发送，把整个数据报当作数据帧的“数据”部分，在前面封装对应的数据链路层协议头“帧头 1”（也可以有 FCS 协议尾），形成数据帧后从网卡通过传输介质在“网络 1”中传输。

【说明】“帧头 1”中封装的源 MAC 地址是源主机网卡的 MAC 地址，目的 MAC 地址是源主机配置的网关接口（即“路由器 1”连接“网络 1”的接口）的 MAC 地址。

（2）当“路由器 1”连接“网络 1”的接口收到来自源主机发来的数据帧时，因为路由器的接口是三层的，需要继续把数据帧上传到该路由器的网络层功能模块，于是要对原来的帧进行解封装，去掉帧头和帧尾信息（在有帧尾的情况下），还原出原来的 IP 数据报，以识别包中的目的地址信息。此时 IP 数据报的内容都没变，仍包括源主机在发送 IP 数据时封装的源和目的 IP 地址信息等。

（3）在“路由器 1”上根据路由器中的路由表信息选择一个出接口（连接“网络 2”的三层接口）进行路由转发。此时又要从网络层向数据链路层传输，进行帧封装，把整个 IP 数据报当作数据帧的“数据”部分，然后加上对应链路的链路层协议头“帧头 2”（也可以有 FCS 协议尾）。

【说明】此时的“帧头 2”与原来的“帧头 1”里面封装的 MAC 地址信息是不一样的。“帧头 2”的源 MAC 地址是“路由器 1”连接“网络 2”的接口的 MAC 地址，目的 MAC 地址是对应路由表中的“下一跳”对应接口的 MAC 地址，即“路由器 2”连接“网络 2”的接口的 MAC 地址。

（4）当从“路由器 1”发送的数据帧传输到“路由器 2”连接“网络 2”的接口时，同样因为该接口是三层的，所以又需要向该路由器的网络层功能模块传输，此时又要对帧进行解封装，去掉帧头和帧尾信息（在有帧尾的情况下），所还原出的仍是原来在源主机上发送的 IP 数据报，使“路由器 2”仍可以识别 IP 数据报中的目的 IP 地址信息。

（5）在“路由器 2”上根据路由器中的路由表信息选择一个出接口（连接“网络 3”的三层接口）进行路由转发。因为目的主机所在网络（网络 3）是直接连接在“路由器 2”上的，所以它可直接根据对应路由表项的出接口把数据从“网络 3”中转发到目的主机上。但首先也要从网络层向数据链路层传输，进行帧封装，把整个 IP 数据报当作数据帧的“数据”部分，然后加上对应链路的链路层协议头“帧头 3”（也可以有 FCS 协议尾），把帧转发到目的主机上。

【说明】此时的“帧头 3”与前面的“帧头 1”和“帧头 2”里面封装的 MAC 地址信息都不一样。“帧头 3”的源 MAC 地址是“路由器 2”连接“网络 3”的接口的 MAC 地址，目的 MAC 地址是目的主机的 MAC 地址。

（6）当从“网络 3”的数据链路层把帧传输到目的主机时，在目的主机上又要对所接收的数据帧进行解封装（去掉帧头和帧尾），还原出原来的 IP 数据报并上传到主机的网络层功能模块，以便目的主机可以对源主机发送的 IP 数据报进行处理。如果在源主机上发送的 IP 数据报是来自最高的应用层，则到达目的主机后也要继续上传到最高的应用层。

从以上可以看出，IP 数据报无论经过了多少个网络，整个数据报内容都是不会改变的，包括

IP 协议头部的源 IP 地址和目的 IP 地址信息。但是 IP 数据报每经过一个 IP 网络向下一个网络发送时都需要经过重新的数据帧封装，此时帧中的源 MAC 地址和目的 MAC 地址都将发生变化。

8.3.8 IPv4 数据报的分段与重组

在网络层中还涉及到一个 IP 数据报分段（或称“分片”）与分段重组的问题。

1. IP 数据报分段

IP 数据报分段发生在发送端，那是因为在发送端网络层的 IP 数据报最终还是要向下面的数据链路层传输，到了数据链路层时又要进行数据帧的封装，把来自网络层的整个 IP 数据报当作帧中的“数据”部分，但是每种数据链路层协议“数据”字段部分的大小是有限制的（如以太网帧中“数据”字段最大为 1500 个字节），也就是一个数据帧可以装下的数据是有限的，这就要求来自网络层的 IP 数据报大小必须在对应数据链路上传输时不能超过该链路层协议帧“数据”字段大小的限制。如果超过了，就需要对网络层发送的原始 IP 数据报进行分段，使每个分段的大小符合对应数据链路层协议“数据”字段大小的限制。

每种数据链路层协议帧“数据”字段的大小也就是不同数据链路的 MTU（最大传输单元），来自网络层的 IP 数据报大小必须小于等于对应链路层协议的 MTU，否则一次性“装”不下，需要分几次来封装，也就是我们这里所说的 IP 数据报分段。表 8-2 列出了几种主要的数据链路层协议的 MTU 值。

表 8-2　不同网络数据链路层协议的 MTU 值

协议	MTU（字节）
FDDI	4352
以太网	1500
PPP	1500
PPPoE	1492
X.25	576

分段后的 IP 数据报，每个分段都要重新进行 IP 协议封装，都包括一个 IP 协议头，然后对每个分段中的“标志”字段和“段偏移”字段进行适当的设置。凡是由同一个 IP 数据报分段的 IP 数据分段中“标志”字段的第 2 位均为 0，除最后一个分段的“标志”字段的最低位均为 0，其余各分段该位均为 1，最高位各 IP 数据分段值均为 0。至于“段偏移”字段，第一个分段的值为 0，其余各分段的值要视具体链路的 MTU 值而定，但必须是 8 个字节的整数倍，而且原始 IP 数据报的上层协议头必须全部包括在第一个分段的“数据”字段中。具体参见 8.3.4 节介绍的 IP 数据报头格式中的相应内容。

在互联网中，包含了各种各样的异构网络，一个路由器也可能连接着具有不同 MTU 值的多个网络，从一个网络上接收的 IP 数据报并不一定能在另一个网络上发送。如图 8-21 所示，一个路由

器连接两个网络，其中一个网络为以太网，根据表 8-2 可知其 MTU 值为 1500 个字节，另一个为 X.25 网络，根据表 8-2 可知其 MTU 值为 576 个字节。



图 8-21 路由器连接不同 MTU 值网络的 IP 数据报传输示例

在本示例中，主机 A 连接着 MTU 值为 1500 的以太网络，这样在以太网上链路层传输的帧中的“数据”部分，也就是 IP 数据报长度不超过 1500 个字节，而主机 B 连接着 MTU 值为 576 的 X.25 网络，这样在 X.25 网上链路层传输的帧中的“数据”部分，也就是 IP 数据报长度不超过 576 个字节，即两个网络中链路层传输的 MTU 值是不一样的。这时如果主机 A 要把一个 1450 个字节的数据报发送给主机 B，路由器可以接收到主机 A 发送的数据报，但却不能在 X.25 网络上转发它，因为它超过了 X.25 网络上链路层 MTU 的限制。

为了解决这个问题，就需要在路由器上对接收到的主机 A 发来的 IP 数据报进行分段（如把 1450 个字节的以太网帧分成 3 段，分别为 576 个字节、576 个字节、298 个字节），然后再将每个分段独立地进行发送，可以像正常的 IP 数据报一样经过独立的路由选择等处理过程，最终到达目的主机。最后，还需要在目的主机上对这些分段进行重组，恢复原始的大 IP 数据报。

这时就涉及到一个问题，IP 数据报分段后是否在每个分段中加上对应的 IP 协议头信息。答案是肯定的，因为分段后的每个 IP 数据分段都需要进行单独的路由，所以必须要有相应的 IP 协议头信息。这时就得把原来 IP 数据报中的包头部分复制到所有由同一个 IP 数据报拆分的分段上。另外，还需要在每个分段的 IP 协议头部分对“标志”字段和“段偏移”字段的值进行对应的修改。

2. IP 数据分段重组

在发送端对一个大的 IP 数据报进行了拆分，分成了多个 IP 数据分段，在接收端自然就要把发送来的一个个分段重新按顺序组合，恢复成原始的大数据报，这就是本节要介绍的另一个问题——“IP 数据分段重组”。

IP 数据分段重组时关键是考虑各个分段中的段偏移值大小，由小到大的顺序进行排列，然后除了第一个分段外，其余各分段的 IPv4 协议头部分要去掉，各分段的“数据”字段部分依次拼接起来就可以还原出原始的 IP 数据报了。不过，最后还要修改第一个分段报头信息，“标志”字段的最低两位分别为 1（不分段）和 0（后面没有分段），“段偏移”字段值为 0。

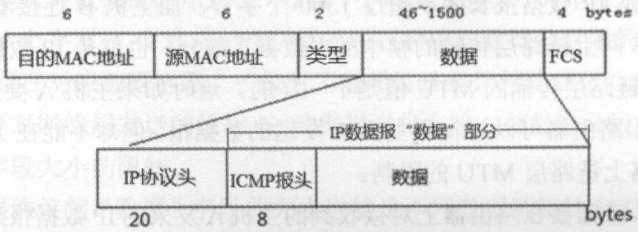
假设有一个 ICMP 类型的 IP 数据报被分成了 3 个 IP 数据分段（不带“选项”字段），然后分别传输到以太网数据链路，最终获知前两个帧的大小均为 1518 个字节，最后一个帧的大小为 86 个字节，求原来在网络层被拆分的各 ICMP 类型 IP 数据分段的偏移值。

因为已知是以太网链路，而以太网帧头和帧尾我们知道一共是 18 个字节，所以要求从网络层下传到数据链路层前各 IP 数据分段的大小时均要减去 18 个字节。由此可以知道，这 3 个 IP 数据段的大小分别是 1500 个字节、1500 个字节和 68 个字节。但这是 ICMP 报文要经过 IP 协议封装并

拆分后的 IP 数据段大小，要求各 IP 数据分段的偏移值最关键的是要得出各 IP 数据分段中“数据”部分的拆分情况。

ICMP 协议报文是要经过 IP 协议封装的，在不带“选项”字段时，IP 协议头部大小为 20 个字节，根据前面得出的各 IP 数据分段大小 1500 个字节、1500 个字节、68 个字节可以很快得出，这 3 个 IP 数据分段中“数据”字段部分的大小分别为 1480 个字节、1480 个字节、48 个字节。又因为偏移是以 8 个字节为单位进行计算的，所以很容易得出第一个 IP 数据报的偏移值为 0，第二个 IP 数据报的偏移值是 186，第三个 IP 数据报的偏移值是 371。

以上计算过程可用图 8-22 来表示。



IP数据报编号	IP数据分段标志	段偏移/偏移字节值	“数据”字段字节大小
1	更多分段位置1	0　0字节	1518-18-20=1480
2	更多分段位置1	186　1480字节	1518-18-20=1480
3	更多分段位置0	371　2960字节	86-18-20=48

图 8-22　IP 数据段偏移计算示例

8.4　ARP 协议

ARP（Address Resolution Protocol，地址解析协议）的全称是 Ethernet Address Resolution Protocol，即以太网地址解析协议。它是通过已知的 IP 地址解析未知的以太网 MAC 地址（或称物理地址）的一种三层协议。但是它的报文是不需要经过 IP 协议封装的（即 ARP 报文头部没有 IP 协议头），因为它比 IP 协议的层次要低，尽管同样是位于网络层。

8.4.1　ARP 报文格式

在网络通信中，无论什么数据最终都要经过数据链路层的“帧封装”后才能在传输介质上进行比特流传输，这就涉及到需要填写帧头的源 MAC 地址和目的 MAC 地址信息。源 MAC 地址好说，就是发送节点的 MAC 地址，但目的节点的 MAC 地址往往是不知道的，因为我们通常只是知道目的节点的 IP 地址，在进行计算机网络应用时填上的也是对方的 IP 地址。要通过已知的目的节点 IP 地址获知目的节点的 MAC 地址就需要依靠 ARP 协议了。它可以通过 ARP 广播报文进行查找。

ARP 协议报文是不需要经过 IP 协议封装的，直接生成自己的报文，当然也包括它自己的协议头（ARP 报头）和数据部分，如图 8-23 所示。下面是各字段的说明。

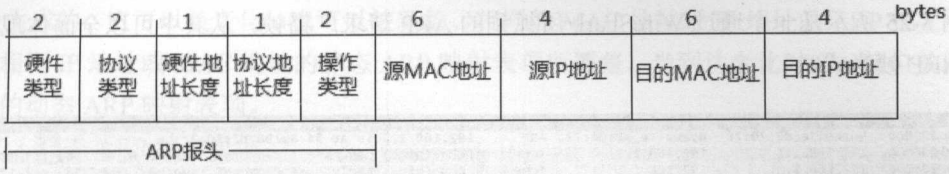


图 8-23 ARP 报文格式

- 硬件类型：2 个字节，标识发送方接口类型，以太网为 0x0001。
- 协议类型：2 个字节，标识网络类型，IPv4 网络为 0x0800。
- 硬件地址长度：1 个字节，以字节为单位的十六进制形式标识 MAC 地址长度，以太网 MAC 地址为 0x06。
- 协议地址长度：1 个字节，以字节为单位的十六进制形式标识 IP 地址长度，固定为 0x04。
- 操作类型：2 个字节，标识 ARP 报文类型，请求报文为 1（0x0001），响应报文为 2（0x0002），均为十六进制形式。
- 源 MAC 地址：6 个字节，标识发送端接口的 MAC 地址。
- 源 IP 地址：4 个字节，标识发送端设备接口的 IP 地址。
- 目的 MAC 地址：6 个字节，表示接收端设备接口的 MAC 地址。在请求报文中该字段值全为 0，即 00-00-00-00-00-00，表示任意地址，因为现在不知道这个 MAC 地址。在响应报文中为真实的接收方设备接口的 MAC 地址。
- 目的 IP 地址：4 个字节，表示接收端设备接口的 IP 地址。

ARP 报文在向下传输到数据链路层时仍需要经过数据链路层协议的帧封装，形成 ARP 帧。ARP 帧就是在图 8-23 所示的 ARP 报文的最前面加上以太网帧头和帧尾，如图 8-24 所示。



图 8-24 ARP 帧格式

以太网帧头和帧尾的 4 个字段说明如下：

- 目的 MAC 地址：6 个字节，如果是 ARP 请求帧，因为它是一个广播帧，所以要填上广播 MAC 地址——FF-FF-FF-FF，其目标是网络上的所有主机；如果是响应帧，则填上接收端接口的 MAC 地址。
- 源 MAC 地址：6 个字节，发送端设备接口的 MAC 地址。
- 类型：2 个字节，标识帧数据部分对应的上层协议类型，因为这是 ARP 协议帧，所以是 ARP 的协议号 0x0806。
- FCS：4 个字节，以太网帧的 CRC 校验码。

如图 8-25 所示是一个通过 WireShark 所抓到的 ARP 请求广播帧，从其中可以全面体现以上所介绍的 ARP 帧格式。

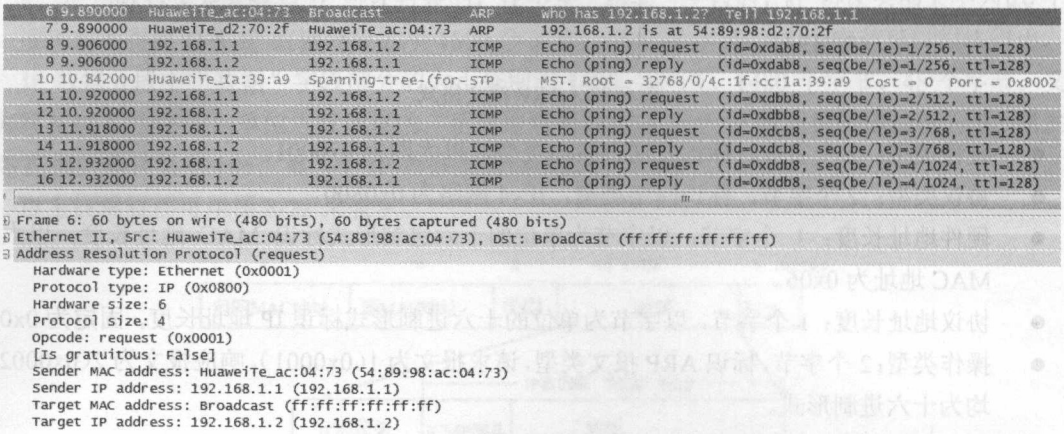


图 8-25　ARP 请求广播帧示例

8.4.2　ARP 映射表

上节介绍到，ARP 协议主要是用来通过已知的 IP 地址解析出对应的未知 MAC 地址，因为双方在初次通信时一般都只知道对方的 IP 地址，不知道对方的 MAC 地址，这样在发送数据时就无法进行帧封装。为了解决这个问题，先通过发送一个 ARP 请求广播帧来查找对方的 MAC 地址，然后再用查找到的 MAC 地址对要发送的数据进行帧封装。但是这里又涉及到一个问题，那就是如果这些查找记录没有保存的话，下次发送数据又不知道对方的 MAC 地址，又要通过 APR 协议来广播查找了，这显然不符合实际的应用需求。

当然这肯定是要有解决方案的，那就是可以在主机或三层设备上创建或者通过自动学习生成 ARP 映射表，把已知的 IP 地址与 MAC 地址关联建立一个映射表，这样下次有数据要发送时直接通过在 ARP 映射表中查看接收方的 IP 地址就可以知道对应的接收方 MAC 地址了，不用再发送 ARP 请求广播报文了。一则频繁发比较麻烦，影响通信效率，二则网络中充满了大量这本来不必要的广播报文，会大量占用宝贵的网络带宽，同时也会大量消耗设备资源的。

ARP 映射表其实我们在第 6 章介绍二层交换时就已经用到了，也需要在主机上创建对应的 ARP 映射表。在三层交换中，ARP 映射表更是不可或缺了，不仅需要在主机上创建，还需要在三层设备上创建，以提高数据转发效率。本节先具体介绍 ARP 映射表中映射表项的分类，下节再具体介绍借助 ARP 映射表实现地址解析的具体原理。

ARP 映射表中的 ARP 映射表项有两类：动态 ARP 映射表项和静态 ARP 映射表项。

（1）动态 ARP 映射表项。

动态 ARP 映射表项是由 ARP 协议通过自动学习所接收到的 ARP 报文中的源 MAC 地址和源

IP 地址生成的。它可以被老化，也可以被新的、相同的 MAC 地址或 IP 地址的 ARP 报文更新，还可能被相同 IP 地址或 MAC 地址的静态 ARP 映射表项所覆盖。当到达老化时间、接口关闭时会删除相应的动态 ARP 映射表项。

（2）静态 ARP 映射表项。

静态 ARP 映射表项是通过管理员手工配置生成的，主机和设备上都会有对应的 ARP 命令来创建。它不会被老化，也不会被动态 ARP 映射表项覆盖。配置静态 ARP 映射表项可以增加通信的安全性，因为静态 ARP 映射表项可以覆盖动态 ARP 映射表项，这样可以防止用户主机非法更改他们的 IP 地址和 MAC 地址（也就是我们通常所说的 IP 地址与 MAC 地址的绑定），此时攻击报文无法修改此表项的 IP 地址和 MAC 地址的映射关系，从而保护了本设备和指定设备间的正常通信。

静态 ARP 映射表项又分为短静态 ARP 映射表项和长静态 ARP 映射表项。

（1）长静态 ARP 映射表项。

在配置长静态 ARP 映射表项时，除了配置 IP 地址和 MAC 地址项外，还必须配置该 ARP 映射表项中 IP 地址、MAC 地址接口所对应的 VLAN 和出接口。也就是长静态 ARP 映射表项同时绑定了 IP 地址、MAC 地址、VLAN 和出接口，可以直接用于报文转发。

（2）短静态 ARP 映射表项。

在配置短静态 ARP 映射表项时，只需要配置 IP 地址和 MAC 地址项。如果出接口是三层以太网接口，短静态 ARP 映射表项可以直接用于报文转发；如果出接口是 VLAN 虚拟接口，短静态 ARP 映射表项不能直接用于报文转发，当要发送 IP 数据报时，先发送 ARP 请求广播报文，如果收到的响应报文中的源 IP 地址和源 MAC 地址与所配置的 IP 地址和 MAC 地址相同，则将接收 ARP 响应报文的接口加入该静态 ARP 映射表项中，之后就可以用于 IP 数据报的转发了。

8.4.3 ARP 地址解析原理

ARP 协议总的地址解析原理很简单，就是源主机通过先发送 ARP 广播请求报文，与请求报文中“目的 IP 地址”一致的主机会做出响应，这样源主机得到目的主机的 MAC 地址。具体又要区分源主机与目的是否在同一 IP 网段两种情形。

1. 源主机与目的主机在同一 IP 网段情形

在如图 8-26 所示的示例中，现假设主机 A 和主机 B 在同一个 IP 网段（均位于 192.168.1.0/24 网段），主机 A 要向主机 B 发送数据。主机 A 和主机 B 的 IP 地址和 MAC 地址均已在图中有标识，现在就是主机 A 已经知道主机 B 的 IP 地址，要想获得主机 B 的 MAC 地址，具体的地址解析过程如下：

（1）主机 A 首先查看自己的 ARP 映射表，确定其中是否包含有主机 B 的 IP 地址对应的 MAC 地址。如果找到了对应的 MAC 地址，则主机 A 直接利用 ARP 映射表中的 MAC 地址对 IP 数据报进行帧封装，并将数据报发送给主机 B。

（2）如果主机 A 在 ARP 映射表中找不到对应的 MAC 地址（初次通信时 ARP 映射表为空），则先缓存要发送的数据报文，然后以广播方式（帧头的“目的 MAC 地址”为广播 MAC 地址——FF-FF-FF-FF-FF-FF，任一同网段的节点均可收到）发送一个 ARP 请求帧。ARP 请求帧中的源 IP

地址和源 MAC 地址分别为主机 A 的 IP 地址（192.168.1.1）和 MAC 地址（0002-6779-0f4c），目的 IP 地址和目的 MAC 地址为主机 B 的 IP 地址（192.168.1.2）和全 0 的 MAC 地址。因为 ARP 请求帧是以广播方式发送，所以该网段上的所有主机都可以接收到该请求包，但只有其 IP 地址与目的 IP 地址一致的主机 B 会对该请求进行处理。

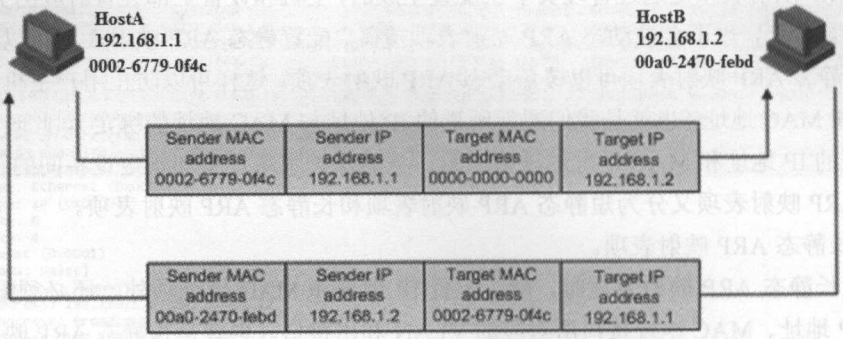


图 8-26　ARP 地址解析示例

（3）主机 B 在收到主机 A 发来的 ARP 请求帧后，基于其中的源 IP 地址和源 MAC 地址建立基于主机 A 的 ARP 映射表项。然后以单播方式向主机 A 发送一个 ARP 响应报文，响应报文中就包含了自己的 MAC 地址，也就是原来在 ARP 请求报文中要请求的目的 MAC 地址。

（4）主机 A 在收到来自主机 B 的 ARP 响应报文后，基于其中的主机 B 的 IP 地地和 MAC 地址创建一个 ARP 映射表项，以用于后续数据的转发，同时将原来缓存的数据经过再次修改（在“目的 MAC 地址”字段填上主机 B 的 MAC 地址）后发送出去。

2. 源主机与目的主机不在同一 IP 网段情形

以上是同一 IP 网段中两主机的 ARP 地址解析的全过程。如果两主机不在同一 IP 网段，它们之间又是如何通信的呢？具体步骤如下：

（1）如果主机 A 不知道网关的 MAC 地址（也就是在主机 A 的 ARP 映射表中没有网关对应的 MAC 地址表项），则主机 A 先在本网段中发出一个 ARP 请求广播报文，ARP 请求广播报文中的“目的 IP 地址”为网关 IP 地址，代表其目的就是想获得网关的 MAC 地址。如果主机 A 已知网关的 MAC 地址，则略过此步。

（2）网关收到 ARP 广播报文后同样会向主机 A 发回一个 ARP 应答报文。当主机 A 收到应答报文后就可以获取网关的 MAC 地址，建立基于网关的 ARP 映射表项。然后在主机 A 向主机 B 发送的数据帧中的“目的 MAC 地址”字段填上网关的 MAC 地址后发给网关。

（3）网关收到主机 A 发来的数据，并且发现其中的“目的 IP 地址”不是自己的，而“目的 MAC 地址”是自己的，就知道这个数据需要网关来转发了。网关首先在自己的 ARP 映射表中查找有关与帧中的“目的 IP 地址”对应的映射表项，如果有则网关直接将来自主机 A 的数据帧中的“目的 MAC 地址”字段填上对应映射表项中的 MAC 地址后转发给主机 B。

（4）如果网关 ARP 映射表中没有与帧中“目的 IP 地址”对应的映射表项，网关会根据对应

的路由表项信息查找目的 IP 地址所在网段的路由路径中对应的出接口上发送 ARP 请求广播报文，以查找所收到的数据帧中“目的 IP 地址”对应的 MAC 地址。此时假设主机 B 是直接连接在网关设备上，因为主机 B 的 IP 地址与 ARP 广播帧中的“目的 IP 地址”一致，所以会接收这个由网关发送的 ARP 请求广播帧并建立基于网关的 ARP 映射表项。然后再以单播方式向网关发回一个响应 ARP 报文，相当于向网关告诉主机 B 自己的 MAC 地址。

（5）当网关收到来自主机 B 的响应报文后，建立基于主机 B 的 ARP 映射表项，然后利用获取的主机 B 的 MAC 地址对收到来自主机 A 的数据重新进行帧封装，此时帧中的源 MAC 地址是网关接口的 MAC 地址，MAC 地址就是主机 B 的 MAC 地址了，然后可以直接发送给主机 B 了。

3. 总结

综上所述，在以上两种情况下 ARP 地址解析的基本思路如下：

（1）当源主机要查找本网段某目的主机的 MAC 地址时，直接发送以目的主机 IP 地址作为帧中“目的 IP 地址”的 ARP 广播请求报文，在本网段内进行广播，这样目的主机就可以收到这个 ARP 广播请求报文，然后进行响应。

（2）如果源主机要查找的目的主机不在本网段，则它不能直接发送以目的主机 IP 地址作为帧中“目的 IP 地址”的 ARP 广播请求报文，因为这个广播报文不能跨网段传播，目的主机根本收不到这个广播报文。此时目的主机 MAC 地址的查找需要通过一级级网关设备发送 ARP 广播请求报文来接力查找。

①源主机以网关 IP 地址作为帧中“目的 IP 地址”（源主机必须要配置好网关 IP 地址）来发送 ARP 广播请求报文，其目的是查找网关 MAC 地址。

②在得到网关的应答后源主机再以网关的 MAC 地址作为帧中“目的 IP 地址”对要发送的数据进行帧封装（其实这是一种欺骗方式，因为“目的 MAC 地址”和“目的 IP 地址”不是对应同一节点的），当网关收到后一看目的 IP 地址并不是自己的，就知道它要继续查找了。

③由网关根据它上面的路由表信息查找目的 IP 地址所在网段的路由路径中对应的出接口上发送 ARP 请求报文，查找与原来从源主机收到的数据帧中的“目的 IP 地址”匹配的目的主机 MAC 地址，此时就相当于替代源主机来查找目的主机的 MAC 地址。

当然，此时第一级网关设备可能还找不到最终的目的主机 MAC 地址，因为目的主机可能不是直接在该级网关设备上，这时第二级网关设备又会从其路由表中查找对应目的 IP 地址所在网段的路由路径的出接口上发送 ARP 广播请求报文，依此类推，直到最终找到目的主机的 MAC 地址。当然这仅是第一次通信时需要这么做，因为经过了第一次后，各设备就会在收到目的主机的应答 ARP 报文后创建基于目的主机的 ARP 映射表项，可直接依据这个 ARP 映射表项进行数据转发了。

8.4.4 免费 ARP

除了以上介绍的 ARP 协议在 MAC 地址解析方面的主要功能，还有几种衍生的 ARP 协议功能，如免费 ARP（Gratuitous ARP）、代理 ARP（Proxy ARP）和逆向 ARP（Reverse ARP，RARP），下面各节将进行简单介绍，本节先来介绍免费 ARP。

免费 ARP 是 ARP 请求广播报文的一个特例，那就是 ARP 报文中的“源 IP 地址”和“目的 IP 地址”是相同的。它的主要作用有 3 个方面：IP 地址冲突检测、MAC 地址更新、防止 IP 地址欺骗。

（1）IP 地址冲突检测。

这要应用于 DHCP 服务器自动为客户机分配 IP 地址的情形，即在客户端收到 DHCP 服务器为它分配的 IP 地址时先会通过向网络中发送一个免费 ARP 报文来验证网络中是否还有其他节点在使用相同的 IP 地址。

ARP 广播请求报文中的“源 IP 地址”和“目的 IP 地址”均为本主机的 IP 地址，“源 MAC 地址”为本主机网卡的 MAC 地址，帧头中的“目的 MAC 地址”为广播 MAC 地址 FF-FF-FF-FF-FF-FF，“数据”部分的“目的 MAC 地址”全为 0。

因为免费 ARP 报文中的源 IP 地址和目的 IP 地址相同，也是以广播方式发送的，这样在网络中如果存在与报文中“目的 IP 地址”相同 IP 地址的节点，肯定会接收这个 ARP 广播报文，并且以单播方式进行响应。如果连续发几次都没有收到响应，则证明网络中没有节点使用 ARP 请求报文中的 IP 地址。

（2）MAC 地址更新。

如果某设备的 MAC 地址信息进行了更新，想要让网络中的各设备上原来所创建的 MAC 地址表和 ARP 映射表做同步更新，最简单的方法就是在这台设备上向网段中发送一个免费 ARP 报文。ARP 广播请求报文中的“源 IP 地址”和“目的 IP 地址”均为本地设备连接本网段接口的 IP 地址，“源 MAC 地址”为设备连接本网段接口更新后的 MAC 地址，帧头中的“目的 MAC 地址”为广播 MAC 地址 FF-FF-FF-FF-FF-FF，“数据”部分的“目的 MAC 地址”全为 0。

这样一来，与该设备在同一 IP 网段中的所有设备都会收到这个 ARP 广播报文，尽管它们不会接收它，但均会从中学习到帧中的“源 IP 地址”和“源 MAC 地址”信息，发现与自己原来在本地设备上创建的 MAC 地址表项或者 ARP 映射表项中同时做同步更新，这样就实现了在网络中同步更新设备 MAC 地址信息的目的。

（3）防止 IP 地址欺骗。

在一些网关攻击中，通常是攻击者仿冒网关的 MAC 地址，但 IP 地址不与网关 IP 地址一样，这样当网络中收到这样的报文后就会误认为网关的 IP 地址发生了变化，会使本地设备上的 MAC 地址项和 ARP 映射表项做同步更新，最终导致本网段内所有用户连接到了非法网关上，可能出现所有用户无法上网的现象。

这时网关设备定期向本网段内发布免费 ARP 报文，就可以在相当程度上预防这种攻击，原理其实与前面介绍的“MAC 地址更新”差不多，只不过这里 ARP 广播请求报文中的“源 IP 地址”和“目的 IP 地址”均为网关自己的 IP 地址，“源 MAC 地址”也为网关真实的 MAC 地址。这样就可以使本网段内的设备纠正原来非法的更新。

8.4.5 代理 ARP

我们知道，广播报文有一个特性，那就是不能跨广播域，即只能在发送该广播报文的主机或节

点对应的 IP 网段内传播，也就不能跨越三层设备来广播。这样就有一个问题了，如果源主机与目的主机在同一个 IP 网段，但它们之间隔了一个或多个三层设备（如图 8-27 所示），此时假设源主机要与目的主机通信，但不知道目的主机 MAC 地址时会在它自己所在的网段内发送以目的主机 IP 地址作为帧中“目的 IP 地址”的 ARP 广播请求报文，显然目的主机会收不到源主机发送的 ARP 广播请求报文了，因为它们是在不同的广播域（三层设备的每个三层端口都是一个广播域）中。



图 8-27　同一 IP 网段主机间隔离一个三层设备的情形示例

此时如果在三层网关设备上开启了代理 ARP 功能则可以解决以上问题。网关在收到源主机发送的 ARP 广播请求报文（是用来查找目的主机 MAC 地址的）后，发现会先以自己的 MAC 地址作为帧中的“源 MAC 地址”，以原来目的主机的 IP 地址作为帧中的“源 IP 地址”对源主机进行应答。

这样源主机就会误认为应答帧中的“源 MAC 地址”就是原来所查找的目的主机的 MAC 地址（实际上是网关的 MAC 地址），因为帧中的“源 IP 地址”是真正目的主机的 IP 地址。然后网关自己再通过在它所连接的其他网段发送 ARP 广播请求报文来查找真正的目的主机 MAC 地址。当源主机以网关的 MAC 地址当作目的主机 MAC 地址向目的主机发送的数据到达网关后，网关重新对数据帧进行封装，帧中的源 MAC 地址是网关自己的 MAC 地址，目的 MAC 地址是网关自己通过 ARP 请求报文得到的真正目的主机的 MAC 地址，IP 地址信息不变，最后发给目的主机。

下面根据图 8-27 的具体情形来分析一下 PC1 是如何获取 PC2（PC1 与 PC2 在同一 IP 网段 192.168.0.0/16）的 MAC 地址的。

（1）PC1 要向 PC2 发起通信时，由于它知道 PC2 的 IP 地址与自己在同一网段，但又不知道 PC2 的 MAC 地址，于是直接发送一个以 PC2 的 IP 地址 192.168.20.200/16 作为帧中“目的 IP 地址”，帧头部分的“目的 MAC 地址”是全 1，即 FF-FF-FF-FF-FF-FF，帧数据部分的“目的 MAC 地址”为 0（代表未知）的 ARP 广播请求帧。这时同在 192.168.0.0/16 网段且与 PC1 没有隔离三层设备的所有节点都可以收到这个 ARP 广播请求报文，包括 PC1 的网关（三层交换机）。

（2）因为 PC2 虽然与 PC1 在同一个 IP 网段，但中间隔离了一个三层交换机，所以它是收不到 PC1 发送的 ARP 广播请求报文的。但 PC1 的网关接口 VLANIF10 能收到这个 ARP 广播请求报文，在三层交换机上启用 ARP 代理功能后，它会以 VLANIF10 接口的 MAC 地址作为帧中的“源 MAC 地址”，以 VLANIF10 接口的 IP 地址（192.168.10.99/24）作为“源 IP 地址”，以 PC1 的 MAC 地址作为帧中的“目的 MAC 地址”，以 PC2 的 IP 地址（192.168.20.200/16）作为帧中的“目的 IP 地址”，向 PC1 以单播形式返回一个 ARP 应答报文。

（3）PC1 在收到网关 VLANIF10 接口发来的应答 ARP 报文后，发现应答报文中的“目的 IP

地址”是 PC2 的 IP 地址，于是它认为帧中的“源 MAC 地址”就是 PC2 的 MAC 地址（实际上是网关 VLANFI10 接口的 MAC 地址）。于是它就直接利用所获得的这个假的 PC2 的 MAC 地址对要发给 PC2 的数据进行封装。

（4）因为帧中的“目的 MAC 地址”实际上是网关 VLANIF10 接口的，所以 PC1 发给 PC2 的数据网关会收到的，但网关看到帧中的“目的 IP 地址”不是自己的，于是它先缓存来自 PC1 的数据，然后替代源主机 PC1 通过发送 ARP 广播请求报文向其他网段查找 PC2 的 MAC 地址。PC2 因为也是直接连接在这台三层交换机上的，所以它可以收到通过 VLANIF20 接口发来的 ARP 广播请求报文，然后做出应答。

（5）PC2 的应答报文到了三层交换机后，就可以从应答 ARP 报文的“源 MAC 地址”字段得到 PC2 的 MAC 地址，然后再从缓存中调出原来 PC1 发给 PC2 的数据进行重新封装，此时帧中的 IP 地址均不变，“源 MAC 地址”换成网关自己的，“目的 MAC 地址”换成 PC2 的 MAC 地址，发给了 PC2，这样就实现了 PC1 向 PC2 发送数据了。PC2 再向 PC1 发送数据就简单了，因为此时在三层交换机上已为 PC1 和 PC2 动态创建了对应的 ARP 映射表项，直接根据对应的映射表项找到出接口进行转发就可以了。

综上所述，代理 ARP 主要用于在同一 IP 网段，但中间却隔离了三层设备的两个物理网段的用户间通信。它的最大优点就是能够在不影响其他三层设备路由表的情况下，在网络上添加一个新的三层设备，并且实现在连接一个与原来某个物理网段在同一 IP 网段的物理网段情形下确保它们之间的三层互通。如华为、H3C 的 Super VLAN 中各子 VLAN 是不能直接进行二层通信的，只能通过三层。但这些子 VLAN 又在同一个 IP 网段，要实现它们之间的三层通信，就必须借助于这里的代理 ARP 功能，具体参见《华为交换机学习指南》。

【注意】要使用代理 ARP，主机上必须不能配置默认网关，否则当查询的 MAC 地址位于本网段之外时，也会把查询任务交给默认网关的，即使目的主机与源主机在同一 IP 网段。

8.4.6 逆向 ARP

RARP（Reverse Address Resolution Protocol，逆向地址解析协议）从其名称就可以看出，它与 ARP 的解析原理是相反的。通过前面的学习我们已经知道，ARP 是通过 IP 地址来解析出对应的 MAC 地址，而 RARP 是通过 MAC 地址来解析出对应的 IP 地址。这常用于无盘工作站，因为无盘工作站所使用的操作系统都是位于服务器端的（无盘工作站自己没有硬盘），在启动时只知道自己网卡的 MAC 地址，其 IP 地址就要通过 RARP 来从服务器端解析出来，因为在服务器端会为每个无盘工作站的 MAC 地址与所分配的 IP 地址建立一个映射表。

如图 8-28 所示，无盘工作站在启动时会向网络中的 RARP 服务器发送一个 RARP 单播请求报文，报文中的“源 MAC 地址”为工作站自己的 MAC 地址，“目的 MAC 地址”为 RARP 服务器的 IP 地址，“源 IP 地址”全为 0（代表未知），“目的 IP 地址”为 RARP 服务器的 IP 地址。

在 RARP 服务器收到工作站发来的 RARP 请求报文后，通过查询本地服务器上的 MAC 地址与 IP 地址映射表得到相应工作站的 IP 地址后会以单播方式对工作站进行响应。RARP 响应报文中的

“源 MAC 地址”为 RARP 服务器的 MAC 地址，“目的 MAC 地址”为工作站的 MAC 地址，“源 IP 地址”为 RARP 服务器的 IP 地址，“目的 IP 地址”为工作站的 IP 地址。这样当工作站收到 RARP 服务器发来的 RARP 响应报文后就知道自己的 IP 地址了（就是响应报文中的“目的 IP 地址”）。

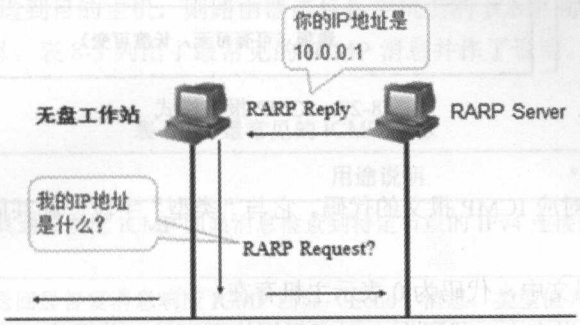


图 8-28 RARP 协议工作原理示意图

【说明】在帧中继网络中称之为 inverse-arp，也可翻译成“逆向 ARP”，但它专用于帧中继网络，是通过已知的二层地址（本地的 DLCI 号码）去查找目的设备的三层 IP 地址。

8.5 ICMP 协议

ICMP（Internet Control Message Protocol，Internet 控制报文协议）是 IPv4 协议簇中的一个子协议，用于在 IP 主机、路由器之间传递控制消息，比 IP 协议的内部层次高。控制消息是指网络通不通、主机是否可达、路由是否可用等网络本身的消息。这些控制消息虽然并不传输用户数据，但是对用户数据的传递起着重要的作用。

ICMP 与 ARP 协议不同，因为 ICMP 在网络层中比 IP 协议的层次高，所以它的报文仍要通过 IP 协议进行封装。它与传输层协议（如 TCP 和 UDP）的目的不同，一般并不用来在端系统之间传送数据，不被用户网络程序直接使用，主要用于像 Ping 和 Tracert 这样的诊断程序的信息回显。

8.5.1 ICMP 报头格式

ICMP 报文封装在 IP 数据报中向下面的数据链路层传输，IPv4 报头在 ICMP 报文的最前面。这样一来，准备向数据链路层传输的 ICMP 报文就包括一个 IP 报头（至少 20 个字节）、一个 ICMP 报头（至少 8 个字节）和一个 ICMP 信息内容（属于 ICMP 报文的“数据”部分）。当 IPv4 报头中的“协议”字段值为 1 时，就说明这是一个 ICMP 报文。ICMP 报头格式如图 8-29 所示，各字段的说明如下：

（1）类型（Type）。

占 1 个字节，标识 ICMP 报文的类型。目前已定义了 14 种，从类型值来看 ICMP 报文可分为两大类：第一类是取值为 1～127 的差错报文，第二类是取值 128 以上的信息报文，具体将在下节中介绍。

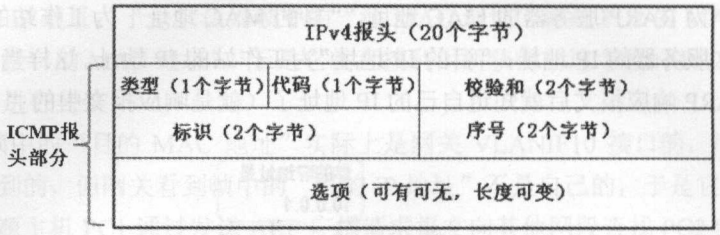


图 8-29　ICMP 报头格式

（2）代码（Code）。

占 1 个字节，标识对应 ICMP 报文的代码。它与“类型”字段一起共同标识了 ICMP 报文的详细类型，具体如下：

- 请求/响应类报文中，代码为 0 表示主机存在。
- 目标不可达类报文中，代码为 0 时表示网络不可达，代码为 1 时表示主机不可达，代码为 2 时表示协议不可达。
- 源抑制类报文中，代码为 0 表示要进行流量控制。
- 超时类报文中，代码为 0 时表示传输超时，代码为 1 时表示重组分段超时。

（3）校验和（Checksum）。

占 2 个字节，是对包括 ICMP 报文“数据”部分在内的整个 ICMP 数据报的校验和，以检验报文在传输过程中是否出现了差错。其计算方法与我们在 8.3.4 节介绍的 IP 报头中的“校验和”计算方法是一样的。

（4）标识（Identifier）。

占 2 个字节，标识本 ICMP 进程，但仅适用于回显请求和应答 ICMP 报文，对于目标不可达的 ICMP 报文和超时 ICMP 报文等该字段值全为 0。

（5）序号（Sequence Number）。

占 2 个字节，用于标识本 ICMP 报文编号。请求和对应的响应报文的“标识”和“序号”字段的值要一致。

如图 8-30 所示是一个 ICMP 回显请求报文示例。其中的 LE 是 little-endian byte order 的简称，翻译为“小端序”，即从低字节开始传输，相当于原序号的第一个字节当成了第二个字节，第二个字节当成了第一个字节。这样，原来为 0000000000000001 就得到 0000000100000000，所以为 256。

```
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0xccd1 [correct]
    Identifier: 0xb9ab
    Sequence number: 1 (0x0001)
    Sequence number (LE): 256 (0x0100)
  Data (32 bytes)
      Data: 08090a0b0c0d0e0f101112131415161718191a1b1c1d1e1f...
      [Length: 32]
```

图 8-30　ICMP 报文示例

8.5.2 ICMP 消息类型

ICMP 报告无法传送的数据报而返回的错误信息，并能帮助对这些错误进行疑难解答。例如，如果 IPv4 数据报不能传送到目的主机，则路由器或目标主机上的 ICMP 协议会向发送主机返回一条“无法到达目标”消息。表 8-3 列出了最常见的 ICMP 消息并作了说明。

表 8-3 最常见的 ICMP 消息

ICMP 消息类型	用途说明
回显请求	Ping 工具通过发送 ICMP 回显消息检查到特定节点的 IPv4 连接来排查网络问题。类型值为 8
回显应答	节点发送回显答复消息响应 ICMP 回显（Echo）消息。类型值为 0
重定向	路由器发送“重定向”消息，告诉发送主机到目标 IPv4 地址的更好路由。类型值为 5
源抑制	路由器发送“源结束”消息，告诉发送主机它们的 IPv4 数据报将被丢弃——因为路由器上发生了拥塞，于是发送主机将以较低的频度发送数据报。类型值为 4
超时	这个消息有两种用途：第一，当超过 IP 报文生存时间时向发送系统发出错误信息；第二，如果分段的 IP 数据报没有在某种时限内重新组合，这个消息将通知发送系统。类型值为 11
无法到达目标	路由器和目标主机发送“无法到达目标”消息，通知发送主机它们的数据报无法传送。类型值为 3

表 8-3 中的“无法到达目标”消息又可以细分，具体如表 8-4 所示。

表 8-4 常见的“无法到达目标”消息

无法到达目标消息	说明
不能访问主机	路由器找不到目标 IPv4 地址的路由时发送“不能访问主机”消息
无法访问协议	目标 IPv4 节点无法将 IPv4 报头中的“协议”字段与当前使用的 IPv4 客户端协议相匹配时会发送“无法访问协议”消息
无法访问端口	IPv4 节点在无法将 UDP 报头中的“目标端口”字段与使用该 UDP 端口的应用程序相匹配时发送“无法访问端口”消息
需要分段但设置了 DF（不分段）	当必须分段但发送节点在 IPv4 报头中设置了“不分段”（DF）标志（置 1 时）时，IPv4 路由器会发送“需要分段但设置了 DF”消息

ICMP 协议只是试图报告错误并对特定的情况提供反馈，但最终并没有使 IPv4 成为一个可靠的协议。ICMP 消息是以未确认的 IPv4 数据报传送的，它们自己也不可靠。

8.5.3 常见的 ICMP 应用

ICMP 协议主要用于 Ping 和 Tracert 操作消息回显，下面分别介绍在这两种操作中 ICMP 的基

本工作原理。

1. ICMP 在 Ping 操作中的应用

Ping 使用一系列 ICMP 的消息来测试目的地址是否可达，是一个发出请求并等待响应的过程。源端首先向目的地址发送 Echo 消息并等待回应，如果 Echo 包到达目的地址并且在确定时间周期内目的地成功返回 Echo Reply 包给源端，则 Ping 成功（如图 8-31 所示），否则会显示 Request timed out 消息。

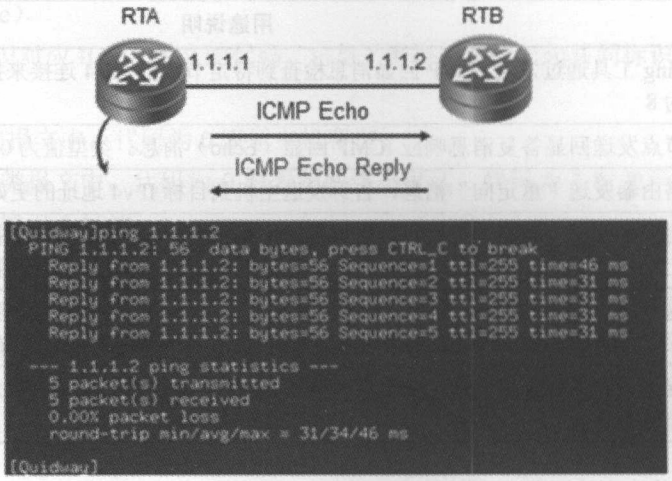


图 8-31　Ping 操作示例

Ping 除基本命令外还提供许多可选参数供使用，这里列举-a 和-i 两个参数。

- -a source-ip-address：设置发送 ICMP Echo-request 报文的源 IP 地址。
- -i interface-type interface-number：设置发送 ICMP Echo-request 报文的接口。

如图 8-30 所示的示例中还可以用 Ping 1.1.1.2 -a 1.1.1.1。

2. ICMP 在 Tracert 操作中的应用

Tracert 是为了探测源节点到目的节点之间数据报文所经过的跃点（Hop），是一项路径跟踪技术。它利用的是 IP 报文的 TTL 字段值在每经过一个路由器（即一个跃点）转发后减 1，当 TTL=0 时则向源节点报告 TTL 超时的机制，具体原理如下（如图 8-32 所示）：

（1）Tracert 首先发送一个 TTL 为 1 的 UDP 报文，随后因为 TTL 超时（因为原始 TTL 为 1，到达第一个跃点后减为 0），因此第一跳发送回一个 ICMP 超时消息（包括对应 Tracert 报文的传输时间、ICMP 消息的返回时间、返回 ICMP 消息的节点 IP 地址）。

（2）Tracert 再发送一个 TTL 为 2 的 UDP 报文，在到达第二个跃点后同样会因为 TTL 超时而返回一个 ICMP 超时消息。

（3）这个过程不断进行（每次发送的 Tracert 报文中的 TTL 值都增加 1），直到到达目的地。Tracert 记录下每一个 ICMP TTL 超时消息的源地址，从而为我们提供了到达目的地所经过的

所有跃点的 IP 地址，也就知道了到达目的地址的整条路径了。

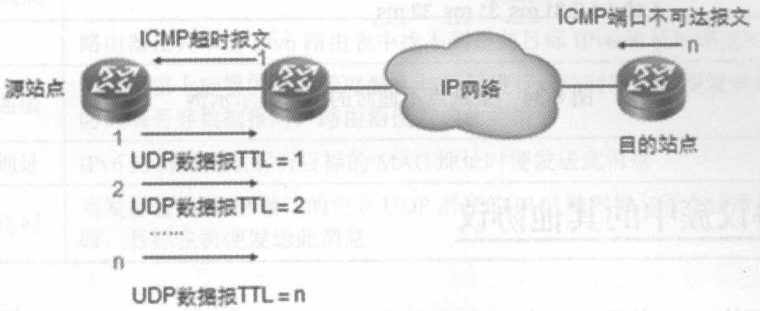
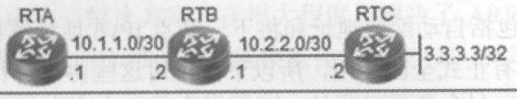


图 8-32 Tracert 工作原理示意图

Tracert 同样可以像 Ping 操作一样对网络连通性进行测试，因为当 Tracert 某一目的地址时，从显示的路径信息可以判断出故障点在什么地方（不通或者跟踪超时的跃点的回显信息为*）。如果路径中某一跳到达不了时则会无法继续测试下一跳，会提前结束 Tracert 操作。如图 8-33 所示是 Ping 操作和 Tracert 操作输出的一个对比图。




```
[Quidway]ping 3.3.3.3
  PING 3.3.3.3: 56  data bytes, press CTRL_C to break
    Reply from 3.3.3.3: bytes=56 Sequence=1 ttl=254 time=63 ms
    Reply from 3.3.3.3: bytes=56 Sequence=2 ttl=254 time=63 ms
    Reply from 3.3.3.3: bytes=56 Sequence=3 ttl=254 time=63 ms
    Reply from 3.3.3.3: bytes=56 Sequence=4 ttl=254 time=63 ms
    Reply from 3.3.3.3: bytes=56 Sequence=5 ttl=254 time=63 ms
```

```
[RTA]tracert 3.3.3.3
 traceroute to  3.3.3.3(3.3.3.3) 30 hops max,40 bytes packet
 1 10.1.1.2 31 ms  31 ms  32 ms
 2 10.2.2.2 62 ms  63 ms  62 ms
```

图 8-33 Ping 操作和 Tracert 操作输出的对比

从图 8-33 可以看出，Ping 操作和 Tracert 操作都可以测试出 RTA 与 RTC 的 3.3.3.3 之间是否连通，但 Ping 操作可以简单直观地知道结果是可达的，而 Tracert 操作则更加详细地显示了数据报的转发路径。数据报首先经过 10.1.1.2，然后经过 10.2.2.2 到达 3.3.3.3。另一方面，使用 Tracert 命令更可以找到问题的故障点。

如以上示例中如果显示信息如图 8-34 所示，则说明数据报可以被送到下一跳 10.1.1.2，但是无法从该路由器再转发出去，证明问题出现在该路由器与目的地之间。

```
[RTA]tracert 3.3.3.3
traceroute to  3.3.3.3(3.3.3.3) 30 hops max,40 bytes packet
1 10.1.1.2 31 ms  31 ms  32 ms
2 ***
```

图 8-34　某跃点不通时的信息显示示例

8.6　IPv6 协议簇中的其他协议

除了前面介绍的 IPv6 协议外，在 IPv6 协议簇中主要还包括 ICMPv6 协议、ND 协议和 MLD 协议。

- ICMPv6（Internet Control Managemet Protocol Version 6，因特网控制信息协议版本 6）协议：取代原来在 IPv4 网络中使用的 ICMP。
- MLD（Multicast Listener Discovery，组播侦听者发现）协议：取代 IPv4 网络中的 IGMP，用于管理 IPv6 多播组成员身份。
- ND（Neighbor Discovery，邻居发现）协议：取代 IPv4 网络中的 ARP 协议，用于管理相邻节点间的交互，包括自动配置地址和将下一跃点 IPv6 地址解析为 MAC 地址。

因为 IPv6 毕竟目前还没有正式全面使用，所以下面仅对这些协议进行简单介绍。

1. ICMPv6 协议

ICMPv6 与前面介绍的 IPv4 协议簇中的 ICMP 一样，用于报告传送或转发中的错误并为疑难解答提供简单的回显服务，最新的是在 RFC 4443 中定义。同时 ICMPv6 协议还为 ND 和 MLD 消息提供消息结构。表 8-5 列出了常见的 ICMPv6 消息类型，表 8-6 给出了已定义的“无法到达目标”消息。

表 8-5　常见的 ICMPv6 消息类型

ICMPv6 消息	说明
回显请求	发送主机发送请求回显消息来检查与特定节点的 IPv6 连接
回显应答	节点发送回显答复消息响应 ICMPv6 请求回显消息
无法到达目标	路由器或目标主机发送“无法到达目标”消息，通知发送主机它们的数据报或有效负载无法传送
数据报太大	路由器发送“数据报太大”消息，通知发送主机数据报因太大而无法转发
超时	路由器发送“超时”消息，通知发送主机 IPv6 数据报的跃点限制已到期
参数问题	路由器在处理 IPv6 报头或 IPv6 扩展报头时如果遇到错误，便发送“参数问题”消息来通知发送主机

表 8-6 常见的“无法到达目标”消息

无法到达目标消息	说明
找不到路由	路由器在其本地 IPv6 路由表中找不到指向目标 IPv6 地址的路由时便发送此消息
管理策略禁止通信	当路由器上配置的某条策略禁止与目标进行通信时路由器便发送此消息。例如，当防火墙丢弃数据报时，路由器便发送此类消息
无法到达目标地址	IPv6 路由器无法解析目标的 MAC 地址时便发送此消息
目标端口无法访问	当发往目标 UDP 端口的包含 UDP 消息的 IPv6 数据报与正在侦听的应用程序不对应时，目标主机便发送此消息

2. ND 协议

虽然说 ND 主要是用来取代 IPv4 网络中的 ARP 协议，但 ND 协议其实是对 IPv4 网络中多个协议的综合升级和改进，如 ARP、ICMP 路由器发现和 ICMP 重定向功能。当然，作为 IPv6 的基础性协议，ND 还提供了其他功能，如前缀发现、邻居不可达检测、重复地址检测、地址自动配置等。

在 IPv4 网络中，ARP 攻击问题已经为广大的网络管理者和设备厂商所认识，ARP 攻击能够造成大面积网络不能正常访问外网，使得正常用户深受其害。针对 ARP 攻击，大部分的网络设备厂商都推出了自己的 ARP 防攻击解决方案，在很大程度上解决了 ARP 攻击的问题。而伴随着 IPv6 网络的建设，IPv6 协议簇中的 ND 协议越来越被重视，而在 ND 协议的设计中与 ARP 协议一样并未提供认证机制，导致网络中的主机是不可信的，从而使针对 ND 协议的攻击非常容易。

ND 协议定义的报文使用 ICMPv6 承载，其类型包括：路由器请求报文、路由器通告报文、邻居请求报文、邻居通告报文和重定向报文，是一组 ICMPv6 消息和过程，用于确定相邻节点间的关系。ND 取代了 IPv4 中使用的 ARP、ICMP 路由器发现和 ICMP 重定向功能，提供了更丰富的功能。

主机可以使用 ND 完成以下任务：

- 发现相邻的路由器。
- 发现并自动配置地址和其他配置参数。

路由器可以使用 ND 完成以下任务：

- 公布它们的存在、主机地址和其他配置参数。
- 向主机提示更好的下一跃点地址来帮助数据报转发到特定目标。

节点（包括主机和路由器）可以使用 ND 完成以下任务：

- 解析 IPv6 数据报将被转发到的一个相邻节点的链路层地址（又称 MAC 地址）。
- 动态公布 MAC 地址的更改。
- 确定某个相邻节点是否仍然可以到达。

3. MLD 协议

MLD 协议用于 IPv6 路由器在其直连网段上发现组播侦听者。组播侦听者（Multicast Listener）是那些希望接收组播数据的主机节点。

路由器通过 MLD 协议可以了解自己的直连网段上是否有 IPv6 组播组的侦听者，并在数据库里做相应记录。同时，路由器还维护与这些 IPv6 组播地址相关的定时器信息。MLD 路由器使用 IPv6 单播链路本地地址作为源地址发送 MLD 报文，使用 ICMPv6 报文类型。所有的 MLD 报文被限制在本地链路上，跳数为 1。

到目前为止，MLD 协议有以下两个版本：

- MLDv1（由 RFC 2710 定义）：源自 IPv4 网络的 IGMPv2。
- MLDv2（由 RFC 3810 定义）：源自 IPv4 网络的 IGMPv3。

所有版本的 MLD 协议都支持 ASM（Any-Source Multicast，任意信源组播）模型；MLDv2 可以直接应用于 SSM（Source-Specific Multicast，指定信源组播）模型，而 MLDv1 需要在 MLD SSM Mapping 技术的支持下才能应用于 SSM 模型。

8.7 路由和路由算法

网络层的主要功能就是将分组从源节点路由到目的节点中，而且在大多数计算机网络中采用的是数据报分组交换方式，数据报文分组需要经过多跳（也就是要经过多少个路由器）才能到达目的地。

“路由”功能其实是一种数据报文的分组交换路径选择行为，是网络层的一种基本功能。“路由”功能与我们旅行或者运输邮件、货物最佳路线的选择（注意，“最佳”并不代表最快，是要综合考虑的）是一个道理。如图 8-35 所示是我们假设要从北京发一封信件到美国的旧金山朋友那里。

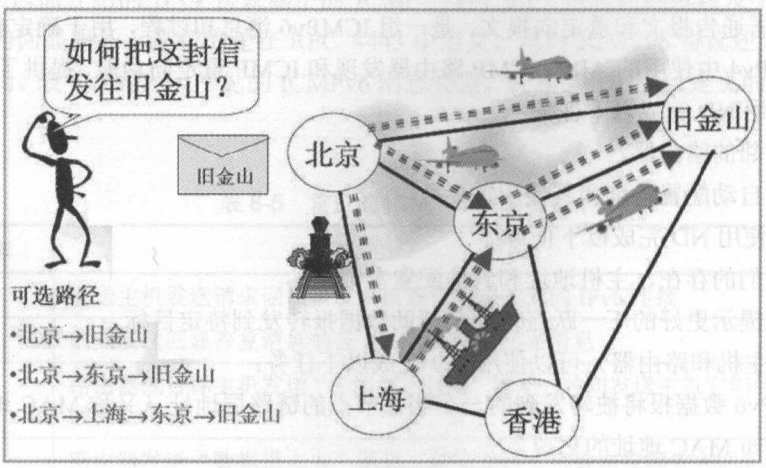


图 8-35　为“路由”功能打的一个比方

从中可以看出，发这样一封信可以有多条路线，不同路线又需要采用不同的邮寄方式、不同的线路长度和邮寄时间，当然运输成本也各不相同，最终选择哪条线路要综合考虑总体邮寄成本、邮

寄时间和途经邮局的可靠性等方面。路由选择也是一样的，它要综合考虑许多因素，如线路长度、信道带宽、线路的稳定性、途经端口的开销等。不同的路由算法所考虑的因素并不相同，本章后面介绍这些路由算法时会有具体介绍。

8.7.1 路由的分类

路由（Routing）是把信息从源节点通过网络传递到目的节点的行为，简单地讲路由就是指三层设备从一个接口上收到数据报，根据数据报的目的地址进行定向并从另一个接口转发出去的过程。但在这条路由路径上至少需要遇到一个中间结点，那就是提供路由功能的设备，如路由器和三层交换机。路由与桥接对比的主要区别在于，桥接发生在体系结构的第二层（数据链路层），连接的是同一网络或同一子网的不同网段，而路由发生在第三层（网络层），连接的是不同网络或不同子网。

路由功能的实现是依靠路由器或三层交换机中的路由表进行的。路由总体上又可以分为：直连路由（Direct Routing）、静态路由（Static Routing）和动态路由（Dynamic Routing）三大类。“直连路由”就是三层设备上直接连接的网段的路由，它不需要配置，也不需要借助其他路由协议，直接生成，而且优先级最高。直接连接在三层设备上的各网段是直接互通的，不需要依靠其他路由。下面分别对“静态路由”和“动态路由”进行介绍。

1. 静态路由

静态路由是我们经常需要配置的，特别是在小型局域网中，因为它配置和管理都比较简单。静态路由中至少包括目的节点或目的网络的 IP 地址和下一跳 IP 地址，或同时包括依据本静态路由转发数据时的出接口，以及其他可选参数。但静态路由仅可根据本静态路由中配置的下一跳将数据报送达到下一跳设备，然后再由其他路由（可以是任意类型路由）一级级地送达到目的节点。

总体来说，静态路由具有以下几个方面的特点：

（1）手动配置。

静态路由需要管理员根据实际需要一条条地自己手动配置，路由器不会自动生成所需的静态路由。这对于比较大的网络来说，全部采用静态路由配置方式工作量就比较大了，而且容易出错。所以静态路由一般用于小型网络或作为大型网络中动态路由的补充。

（2）路由路径固定不变。

因为静态路由是手动配置的、静态的，所以当网络的拓扑结构或链路的状态发生变化时，这些静态路由不能自动修改，需要网络管理员手动去修改路由表中相关的静态路由信息。

（3）不可通告性。

静态路由信息在默认情况下是私有的，不会通告给其他路由器，也就是当在一个路由器上配置了某条静态路由时，它不会被通告到网络中相连的其他路由器上。但网络管理员还是可以通过重发布静态路由为动态路由类型使得网络中的其他路由器也可获此静态路由。

（4）单向性。

静态路由是具有单向性的，也就是它仅为数据提供沿着下一跳的方向进行路由，不提供反向路

由。所以如果你想要使源节点与目的节点或网络进行双向通信，则必须同时配置回程静态路由。在与读者朋友的交流中经常发现这样的问题，就是明明配置了到达某节点的静态路由，可还是 Ping 不通，其中一个重要的原因就是没有配置回程静态路由。

如图 8-36 所示，如果想要使 PC1（PC1 已配置了 A 节点的 IP 地址 10.16.1.2/24 作为网关地址）能够 Ping 通 PC2，则必须同时配置以下两条静态路由，具体配置方法在此不作介绍：

①在 R1 路由器上配置到达 PC2 的正向静态路由（以 PC2 所在网络 10.16.3.0/24 作为目的网络，以 C 节点 IP 地址 10.16.2.2/24 作为下一跳地址）。

②在 R2 路由器上配置一条到达 PC1 的回程静态路由（以 PC1 所在网络 10.16.1.0/24 作为目的网络，以 B 节点 IP 地址 10.16.2.1/24 作为下一跳地址）。

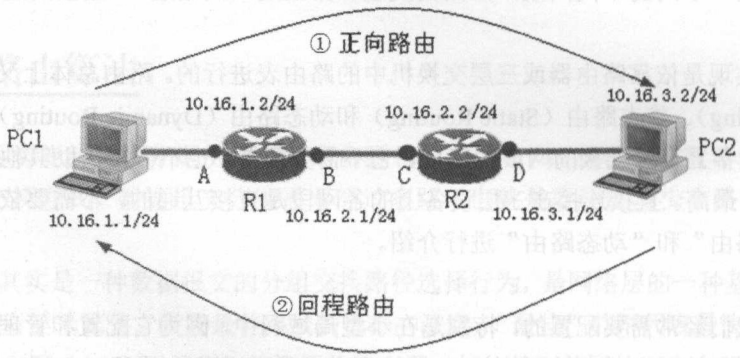


图 8-36　静态路由单向性示例

（5）接力性。

如果某条静态路由中间经过的跳数大于 1（也就是整条路由路径经历了 3 个或以上路由器结点），则必须在除最后一个路由器外的其他路由器上依次配置到达相同目的节点或目的网络的静态路由，这就是静态路由的“接力”特性，否则仅在源路由器上配置这个静态路由则还是不可达的。

就像你要从长沙到北京去，假设中间要途经的站点包括：武汉、郑州、石家庄，可人家只告诉你目的地是北京，以及从长沙出发的下一站是武汉。对于一个没有多少旅游经验的人来说，你是不可能知道到了武汉后又该如何走的，必须有人告诉你到了武汉后再怎么走，到了郑州后又该怎么走，……。这就是“接力性”。

如图 8-37 所示是一个 3 个路由器串联的简单网络，各个路由器节点及 PC 机的 IP 地址均在图中进行了标注，PC1 已配置好指向 R1 的 A 节点地址的网关，现假设要使 PC1 能 Ping 得通 PC2，则需要配置以下 4 条路由（两条正向，两条回程）：

①在 R1 路由器上配置到达 PC2 的正向静态路由（以 PC2 所在网络 10.16.4.0/24 作为目的网络，以 C 节点 IP 地址 10.16.2.2/24 作为下一跳地址）。

②在 R2 路由器上配置到达 PC2 的正向接力静态路由（同样以 PC2 所在网络 10.16.4.0/24 作为目的网络，以 E 节点 IP 地址 10.16.3.2/24 作为下一跳地址）。

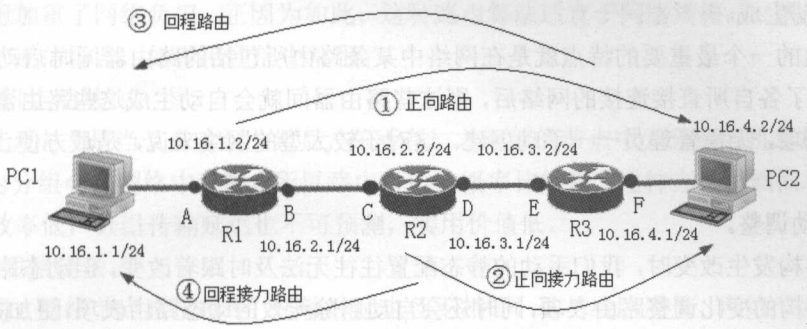


图 8-37 静态路由接力性示例

③在 R3 路由器上配置一条到达 PC1 的回程静态路由（以 PC1 所在网络 10.16.1.0/24 作为目的网络，以 D 节点 IP 地址 10.16.3.1/24 作为下一跳地址）。

④在 R2 路由器上配置一条到达 PC1 的回程接力静态路由（同样以 PC1 所在网络 10.16.1.0/24 作为目的节点地址，以 B 节点 IP 地址 10.16.2.1/24 作为下一跳地址）。

（6）迭代性。

许多读者一直存在一个错误的认识，那就是认为静态路由的“下一跳”必须是与本地路由器直接连接的下一个路由器接口。前面说了，静态路由没有建立邻接关系的 Hello 包，静态路由也不会被通告邻居路由器，所以它的下一跳纯粹是由配置的“下一跳 IP 地址”直接指定或者通过配置的“出接口”间接指定。理论上来说，静态路由的下一跳可以是路径中其他路由器中的任意一个接口，只要能保证到达下一跳即可。这就是静态路由的“迭代性”。

如图 8-37 所示的网络中，如果要在 R1 上配置一条到达 R3 所连接的 10.16.4.0/24 的静态路由。按照正常思维的话，其下一跳应该是 R2 的 C 接口。不过，其实也可以是 R2 的 D 接口或者 R3 的 E 接口或 F 接口。只要通过其他路由能到达这些接口，则这条静态路由就是成功的。

（7）适用于小型网络。

静态路由一般适用于比较简单的小型网络环境，因为在这样的环境中，网络管理员易于清楚地了解网络的拓扑结构，便于设置正确的路由信息。同时小型网络所需配置的静态路由条目不会太多。如果网络规模较大，拓扑结构比较复杂，则不宜采用静态路由，因为这样的配置工作量实在太大。

2. 动态路由

静态路由对于小型且变化不是很频繁的网络来说还是可行的，如局域网。但是对于较大型的广域网，由于拓扑结构较复杂，且网络结构可能经常变动，静态路由就不再适用了（配置工作量太大，且容易出错），通常是采用更加灵活更具自动特性的动态路由了。

动态路由的类型有很多，如 EIGRP（Cisco 的私有路由协议）、RIP、OSPF、IS-IS、BGP 等，不同动态路由协议有不同的特点和主要的适用场景，具体参见《Cisco 路由器配置与管理完全手册》（第二版）《H3C 路由器配置与管理完全手册》（第二版）或《华为路由器学习指南》。

总体来说，动态路由具有以下几个方面的特点：

（1）自动生成。

动态路由的一个最重要的特点就是在网络中某条路由所包括的路由器同时启动了某种动态路由协议，通告了各自所直接连接的网络后，则这些路由器间就会自动生成这些路由器直接连接的网络间的路由表项，无需管理员一一手动创建。这对于较大型的网络来说，是最方便、最简单的路由选择了。

（2）自动调整。

当网络结构发生改变时，我们手动的静态配置往往无法及时跟着改变，但动态路由可以随时根据网络拓扑结构的变化调整路由表项，同时还会自动删除无效的动态路由表项，更加方便路由管理。

（3）自动通告。

前面说了，一个路由器上的静态路由表项是一个路由器私有的，但是动态路由可以在相邻路由器上相互通告，以便及时反映拓扑结构的变化，生成新的动态路由表项。

（4）自动生成双向路由。

虽然单条动态路由也是单向的，但是路由器在生成某条路径的动态路由时会自动生成回程路由表项，也就是会同时生成双向路由表项。

8.7.2 路由算法基础

路由算法（Routing Algorithm）是在给定一组路由器及连接路由器链路的情况下，找出一条从源到目标的最佳路径的方法。通俗地讲，就是把路由器选择最佳路径的策略称为路由算法，是路由器的关键所在。为了完成这项工作，在路由器中需要收集和保存各种与传输路径相关的数据，如拓扑结构、端口度量、端口速率等，然后根据相应的路由算法生成一个个路由表（RoutingTable）表项或其他信息库，如 OSPF、IS-IS 中的 LSDB（链路状态数据库），供数据报转发时提供路由选择。

从大的方面来说，路由算法可以分为两类：非自适应路由算法（Nonadaptive Algorithm）和自适应路由算法（Adaptive Algorithm）。非自适应路由算法的典型代表就是我们常说的静态路由，但早期还包括其他类型的路由。而动态路由中所采用的算法都属于自适应路由算法。

1. 非自适应路由算法

非自适应路由算法是指那些不能根据网络拓扑结构的变化而更新路由表，仅使用静态路由表的路由算法，也称为“固定式路由选择算法”。非自适应路由算法的特点是简单、开销小、灵活性差。它主要包括：静态路由（Static Routing）、扩散（Flooding）法、随机走动（Random Walk）法和最短路径（Shortest Path）法。

（1）静态路由。

“静态路由”一般不认为是真正的路由算法，也就是我们在 8.7.1 节中介绍的“静态路由”。静态路由表项是由管理员手动创建的。

（2）扩散法。

“扩散法”是当一个路由器接口收到一个报文分组后，即向它所有的接口（包括接收该分组的源接口）进行复制扩散。因为有多条可能的路径，所以即使网络局部出现了故障也不影响通信，但

大量重复分组加重了网络负担。正因为如此，这种路由算法适宜于网络规模小、通信负载轻、可靠性要求极高的通信场合，如军用通信中常用。

（3）随机走动法。

“随机走动法”是当节点收到分组后，在所有与之相邻的节点中为分组随机选择出一个节点转发出去。因为分组会在网络中乱窜，所以路由可达的概率比较高。这种方法虽然简单，但不是最佳路由，通信效率低，分组传输延迟也不可预测，实用价值低。

（4）最短路径法。

一般来讲，网络节点直接相连，传输时延也各不相同，这与线路质量、网络节点“忙”与“闲”的状态、节点处理能力等很多因素有关。在定量分析中，常用“开销最小”作为网络节点之间的选择依据，节点间的传输时延是决定费用的主要因素。

“最短路径法”是由 Dijkstra 提出的，其基本思想是：将源节点到网络中所有节点的最短（是指通信链路的长短，不根据开销来计算）通信路径都找出来，作为这个节点的路由表。当网络的拓扑结构不变、通信量平稳时，该点到网络内任何其他节点的最佳路径都在它的路由表中。如果每一个节点都生成和保存这样一张路由表，则整个网络通信都在最佳路径下进行。每个节点收到分组后，查表决定向哪个后继节点转发。

2. 自适应路由算法

我们通常所说的动态路由均属于“自适应路由算法”类型，可根据网络流量和拓扑结构的变化更新路由表。特点是开销大，健壮性和灵活性好。在这类路由算法中，路由器又是如何收集网络拓扑结构和端口信息，然后确定最佳路由的呢？这要依据不同的动态路由协议所采用的动态路由算法类型而定。

总体来说，这些动态路由协议中主要有两种动态路由算法：总体式路由算法和分散式路由算法。采用分散式路由算法时，每个路由器只与直接相连的路由器交换路由信息，其代表就是“距离矢量”（Distance Vector，DV）路由算法。而采用总体式路由算法时，每个路由器都拥有网络中某个区域或者整个网络中所有其他路由器的全部信息以及网络的流量状态，其代表包括“链路状态”（Link State，LS）路由算法和“分级路由”（Hierarchical Routing）路由算法。

（1）距离矢量路由算法。

“距离矢量”简单地讲就是有方向的距离，也就是对应路由条目中源和目的站点间的距离。这个“距离”最初就是指“跳数”（Hop），或者叫“度量”（Metric），从本结点起，每经过一个路由器（也可以是提供路由功能的三层交换机，下同）就加一跳。也就是对应路由条目中除源站点所在网络直接连接的路由器外，到达目的站点所经过的路由器数（路径中所有路由器数减 1）。

如图 8-38 所示是一个 4 台路由器串联的网络示例，则从 PC1 到达 PC2 的路由条目的跳数（也就是距离）为 3（共经过 4 个路由器，减 1 后得到 3）；但是 PC3 到达 PC2 的跳数就只有 1 了（共经过 2 个路由器，减 1 后得到 1），因为它们两个之间除了源路由器外，只经过了一个路由器。在这里要特别注意的是，跳数是针对所经过的路由器数而言的，而不是所经过的网络数而言的，因为连接在同一路由器上的各个网络之间是没有跳数的，亦即跳数为 0。

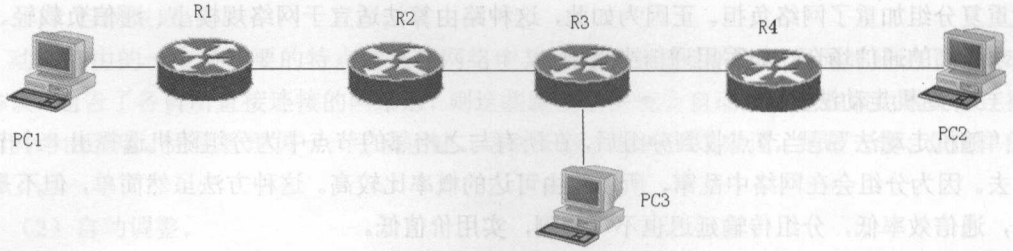


图 8-38　距离矢量计算示例

采用“距离矢量”路由算法的典型路由协议就是 RIP，但它规定一条 RIP 路由中最多只能有 15 跳，超过这个数就认为路由不可达。其他采用“距离矢量”路由算法的路由协议包括 IGRP（内部网关路由协议）、EIGRP（增强型内部网关路由协议）和 BGP（边界网关协议）。EIGRP 同时还支持“链路状态”路由算法，BGP 中的“距离”是指所经过的 AS 系统数而不是所经过的路由器数。

（2）链路状态路由算法。

“链路状态”路由算法（也叫做“最短路径优先”路由算法）比较复杂，它不仅要根据所经过的路由器多少，还要根据路径中各段链路的状态来计算最佳路由路径。而这个“链路状态”包括：接口的IP地址和子网掩码、网络类型(如以太网链路或串行点对点链路)、该链路的端口开销(Cost)、该链路上所有的相邻路由器。而端口开销又与端口的接入带宽有关。

采用“链路状态”路由算法的动态路由协议主要包括：OSPF（开放最短路径优先）、IS-IS（中间系统到中间系统）、EIGRP（同时支持“链路状态”和“距离矢量”两种路由算法）。

总体而言，“链路状态”路由协议是层次式的，网络中的路由器并不向邻居路由器传递“路由表项”，而只是向邻居路由器通告它的一些链路状态。在这类路由协议网络中，会把网络中的所有路由器分成区域，收集区域的所有路由器的链路状态信息，然后根据状态信息生成网络拓扑结构，最后每一个路由器再根据拓扑结构计算出路由。而“距离矢量”路由协议是平面式的，所有的路由学习完全依靠邻居路由器，交换的是完整的路由表项，所以“链路状态”路由协议更灵活，更适合大型网络。

以上介绍的“距离矢量”和“链路状态”路由算法其实只是针对一个特定的区域而言，在像OSPF、IS-IS 这样的大型路由协议中还有一种是针对整个网络中不同区域的路由算法（即区域间的路由算法），即“分级路由”路由算法。

（3）分级路由路由算法。

在前面介绍的“距离矢量”和“链路状态”路由算法中，每个路由器都需要保存其他路由器的一些信息。随着网络规模的扩大，网络中的路由器也将增加。因此，路由表的规模将增大，从而使路由器不能有效地处理网络流量。使用“分级路由”(Hierarchical Routing）路由算法可以解决这个问题。

在“分级路由”路由算法中，路由器被分成很多组，称为区域。每个路由器都只有自己所在区域路由器的信息，而没有其他区域路由器的信息。所以在其路由表中，路由器只需要存储其他每个

区域的一条记录。采用分级路由路由算法的典型路由协议包括 OSPF 和 IS-IS，在区域间是采用“分级路由”路由算法。

以上这些路由算法的工作原理将在后面具体介绍。

8.7.3 路由表基础

路由表（Routing Table）是一个存储在路由器或者联网计算机中的电子表格（文件）或类似的数据库。路由表存储着指向特定网络地址的路径和度量值，可使路由器根据路由表信息计算出到达网络周边的拓扑信息。

路由表建立的主要目的是为了路由选择，路由器通常依靠所建立及维护的路由表来决定对具体数据如何转发。无论是静态路由还是各种动态路由，都会在路由器的路由表中保存，但是不同类型的路由表所显示的信息不完全相同。由设备管理员事先手动创建的固定路由表称为“静态路由表”，不会随未来网络结构的改变而改变；动态路由表是路由器根据所选择的动态路由协议算法自动生成的。

需要特别注意的是，除了前面所说的静态路由和动态路由外，还有一种路由，即直连路由，就是在路由器上直接连接的网络。因为连接在同一个路由器上的各个网络都是相通的，无需配置任何其他路由条目，只需要在路由器上进行通告即可。

在路由器路由表的显示中，不同路由的代码（Codes）是不一样的。在华为设备上只需要执行 display ip routing-table 命令即可显示当前路由器路由表中的所有 IP 路由表项。

```
<Huawei> display ip routing-table
Route Flags: R - relay, D - download to fib
------------------------------------------------------------------------------
Routing Tables: Public
         Destinations : 14        Routes : 14

Destination/Mask    Proto   Pre  Cost      Flags NextHop         Interface

        0.0.0.0/0   Static  60   0           RD  10.137.216.1    GigabitEthernet 2/0/0
       10.10.10.0/24  Direct  0    0           D   10.10.10.10    GigabitEthernet 1/0/0
     10.10.10.10/32  Direct  0    0           D   127.0.0.1      InLoopBack0
   10.10.10.255/32  Direct  0    0           D   127.0.0.1      InLoopBack0
       10.10.11.0/24  Direct  0    0           D   10.10.11.1     LoopBack0
     10.10.11.1/32  Direct  0    0           D   127.0.0.1      InLoopBack0
   10.10.11.255/32  Direct  0    0           D   127.0.0.1      InLoopBack0
     10.137.216.0/23  Direct  0    0           D   10.137.217.208 GigabitEthernet 2/0/0
  10.137.217.208/32  Direct  0    0           D   127.0.0.1      InLoopBack0
  10.137.217.255/32  Direct  0    0           D   127.0.0.1      InLoopBack0
      127.0.0.0/8   Direct  0    0           D   127.0.0.1      InLoopBack0
      127.0.0.1/32  Direct  0    0           D   127.0.0.1      InLoopBack0
127.255.255.255/32  Direct  0    0           D   127.0.0.1      InLoopBack0
255.255.255.255/32  Direct  0    0           D   127.0.0.1      InLoopBack0
```

从以上 IP 路由表信息可以看出，IP 路由表中包含了以下字段：

- Destination：表示此路由的目的地址，用来标识 IP 数据报的目的地址或目的网络。
- Mask：表示此目的地址的子网掩码长度，与目的地址一起来标识目的主机或目的网络所在的网段地址。
- Proto：表示学习此路由的路由协议，包括静态路由（Static）、直连路由（Direct）和各种动态路由。
- Pre：Preference，表示此路由的路由协议优先级，这是用来比较不同协议类型、相同目的地址的多条路由的优先级。同一目的地可能存在不同下一跳、出接口等的多条路由，这些不同的路由可能是由不同的路由协议发现的，也可以是手工配置的静态路由。优先级高（数值小）者将成为当前的最佳路由。
- Cost：路由开销，这用来比较同一种路由协议类型、相同目的地址的多条路由的优先级。但不同类型协议路由的开销类型不同，如距离矢量类协议采用的是“距离”即“跳数”作为路由开销，而链路状态类协议采用“链路状态”（由链路带宽、网络传输性能等参数共同决定）作为路由开销。当到达同一目的地的多条路由具有相同的路由优先级时，路由开销最小的将成为当前的最佳路由。
- FlagsNextHop：表示此路由的下一跳 IP 地址，指明数据转发路径中的下一个三层设备。
- Interface：表示此路由从本地设备发出的出接口。

8.7.4 路由协议优先级

对于相同的目的地，不同的路由协议（包括静态路由）可能会发现不同的路由，但这些路由并不都是最佳的。事实上，在某一时刻到某一目的地的当前路由仅能由唯一的路由协议来决定。为了判断最佳路由，各路由协议都被赋予了一个优先级（在 Cisco 设备中称之为“管理距离”），当存在多个路由信息源时，具有较高优先级（取值较小）的路由协议发现的路由将成为最佳路由，并将最佳路由放入 IP 路由表中。

Cisco 设备中不同路由协议的默认管理距离如表 8-7 所示，华为/H3C 设备中不同路由协议的默认优先级如表 8-8 所示。

表 8-7 Cisco 设备不同路由协议的默认管理距离

路由类型	默认管理距离	路由类型	默认管理距离
直连路由	0	OSPF	110
静态路由	1	IS-IS	115
EIGRP 汇总路由	5	RIP	120
外部 BGP	20	EGP	140
EIGRP	90	外部 EIGRP	170
IGRP	100	内部 BGP	200

表 8-8 华为/H3C 设备不同路由协议的默认优先级

路由类型	默认优先级	路由类型	默认优先级
直连路由	0	OSPF ASE（外部 AS 系统）	150
OSPF	10	OSPF NSSA（非纯末梢区域）	150
IS-IS	15	IBGP	255
静态路由	60	EBGP	255
RIP	100	未知路由	256

以上仅是默认的优先级设置，管理员可以将每条路由修改为不同于默认值的优先级值。通过对到达同一目的节点（或目的网络）的多条相同类型的路由配置不同的优先级值（在静态路由中称之为“浮动静态路由”）就可以达到“热备切换”的目的。就是当优先级较高的当前路由失效时，优先级较低的同一目的地路由马上可以接替原来路由的工作，以确保到达某一目的地的路由始终有效。

【说明】以上所说的路由协议优先级是用于当设备 IP 路由表中有多条可到达相同目的网络（路由表项中的网络 IP 地址子网掩码长度都一样）时所进行的比较，优先级高（优先级值越小，优先级越高）的最终被选择。但我们前面已经知道，在 IP 数据报中并没有“目的 IP 地址”对应的子网掩码字段，所以在 IP 数据报进入设备后可能有多条不同的路由表项（路由协议类型可以相同，也可以不同）与 IP 数据报中的“目的 IP 地址”匹配，这时就不能依据上面介绍的路由协议优先级来选择了，而是要根据“最长匹配规则”进行选择。即同时匹配 IP 数据报“目的 IP 地址”的路由表项中，子网掩码长度最长的路由表项最终被选择。

8.7.5 路由算法的设计目标和设计考虑

路由算法通常具有下列设计目标中的一个或多个：正确性（Correctness）、简单性（Simplicity）、健壮性（Robustness）、稳定性（Stability）、公平性（Fairness）、最优性（Optimality）、快速收敛性（Fast Convergence）和灵活有效性（Efficiency）。

1. 正确性和简单性

对于正确性和简单性比较好理解，就是要求所设计的路由算法能正确计算出网络中的最优路径，而且还要求是最简单的，太复杂的会严重消耗路由器资源。换句话说，路由协议必须高效地提供其路由功能，尽量减少软件和应用的开销。当实现路由算法的软件必须运行在物理资源有限的计算机上时高效尤其重要。

2. 健壮性和稳定性

健壮性和稳定性就是要求路由算法在出现某些设备工作不正常或不可预见软硬件事件的情况下必须仍能正常处理，不会产生大的网络震荡。因为网络一旦投入运行，特别是大型公共服务网络，会要求持续、有效运行数年，甚至更久。在此期间，将会有各种各样的软硬件问题。主机、路由器和线路都有可能会经常性的失败，网络拓扑结构也可能会多次发生变化。路由算法应能很好地适应

这些变化，实现快速的收敛，而且不会要求所有主机都停止工作，即使在某台路由器崩溃时。

3. 公平性和最优性

公平性是要求路由算法对网络中的所有用户必须是平等的；最优性指路由算法选择最佳路径的能力，根据 Metric 的值和权值来计算。例如有一种路由算法可能使用跳数和延迟，但可能延迟的权值要大些。当然，路由协议必须严格定义计算 Metric 的算法。表面上来看这两者是有些冲突的，但事实上一个好的路由算法必须在两个指标之间选择一个最佳的平衡点。

4. 快速收敛性

快速收敛性是指路由算法必须能快速聚合，聚合是所有路由器对最佳路径达成一致的过程。当某网络事件使路径断掉或不可用时，路由器通过网络分发路由更新信息，促使最佳路径的重新计算，最终使所有路由器达成一致。聚合很慢的路由算法可能会产生路由环或网路中断。

在如图 8-39 所示的简单路由结构中，某分组 X 在时间 t1 到达路由器 1，路由器 1 已经更新，并知道到达目的地的最佳路径是以路由器 2 为下一跳，于是就把该分组转发给路由器 2。但是如果由于没有快速链路聚合性能，路由器 2 则可能还没有及时更新，它仍采用它上次数据发送的路由表，认为它的最佳下一跳是路由器 1，于是把该分组发回给路由器 1，结果分组在两个路由器间来回传递，直到路由器 2 收到路由更新信息或分组超过了生存期。

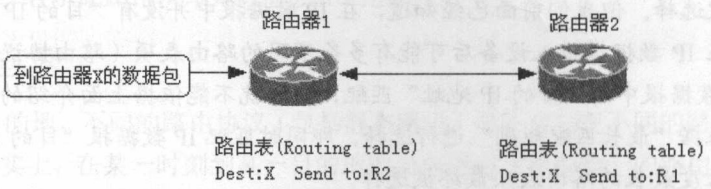


图 8-39 路由器不能快速聚合链路的示例

5. 灵活有效性

路由算法还应该是灵活有效的，即它们应该迅速、准确地适应各种网络环境。例如，假定某网段断掉了，当知道问题后，很多路由算法对通常使用该网段的路径迅速选择次佳的路径。路由算法可以设计得适应网络带宽、路由器队列大小和网络延迟。

在设计路由算法时，还需要考虑许多路由技术元素，如性能标准、成本、决策时间、决策地点、网络信息源和更新时间等。就性能标准方面需要考虑：跳数、成本、延迟、吞吐量等几个方面。成本与数据率有关（数据率越高，成本越低），成本也与当前排队延迟有关。可采取最小跳计数标准或者最小成本标准。

在“决策时间”方面，对于内部数据报，需要为每个包单独作路由决策；对于内部虚电路，只需在建立虚电路时作路由决策。在“决策地点”方面，对于分布式路由方式，每个节点都负责为到达的包选择一条输出链路，需要考虑每个节点；对于集中式路由方式，由某些指定节点（如网络控制中心）负责决策，所以只需针对中心节点；对于源路由方式，路由决策实际上由源站点而非网络作出，所以只需针对原始节点。

在“网络信息源”和“更新时间”这两个方面，大多数路由决策要求根据网络拓扑、交通负载和链路成本等知识作出决定。“更新时间”是信息源和路由决策的函数。如果像扩散路由算法之类是无信息的，也就无更新之说；如果信息是局部的，则更新基本上是连续的；如果是邻接节点或者全部节点，则在采用非自适应算法时从不更新信息，而在采用自适应算法时则要经常更新信息。

8.7.6 深入理解路由与网关的区别

说到路由就不得不说另一个具有与之极其相似功能，却令许多读者朋友绞尽脑汁都想不明白它们之间区别的“网关”概念。

我们知道路由是三层功能，网关也是三层功能，而且它们都可以为三层数据包提供转发路径。这就带来了一个问题，那就是为什么有了路由还要网关，或者反之，真有点“既生瑜，何生亮”的感慨。其实这两者只是在功能上有相似之处，但区别很大。

路由和网关最终的功能区别是：路由可以实现直连或者非直连网络间的三层互通，网关只能实现直连网络间的三层互通，而且只是“一跳”（即一个网络）的连接。由此可以看出，就功能上来说相同之处就是路由和网关都可以实现直连网络间的三层互通，但实现的原理不一样。而且网关不能实现非直连网络间的三层互通。

1. 路由与网关的区别

首先来说路由。路由的数据转发功能依据的是各种路由协议为达到不同目的网络所建立的路由表项，提供了从源网络到达目的网络的完整转发路径（可以是直连网络间，也可以是非直连网络间，非直连网络时要使用动态路由）。

再来看什么是网关。网关可以简单地理解为相邻网络之间隔的那扇“门”。我们知道，直接相邻的两个房间之间都有一扇门，用来相互进入，门的两面分别位于所连接的两个房间。但这扇门只能用于相邻房间的进入，而不能进入到非相邻房间。在图 8-40 中，从房间 A 到房间 B 或者从房间 B 到房间 A 可以直接通过门 1 进入；同理，从房间 B 到房间 C 或者从房间 C 到房间 B 可以直接通过门 2 进入。但是却不能通过门 1 从房间 A 直接进入到房间 C，同样也不能通过门 2 从房间 C 直接进入到房间 A，因为房间 A 与房间 C 是非直接相邻的房间。

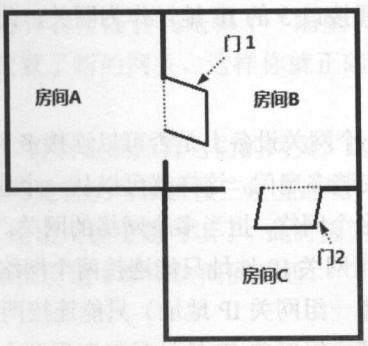


图 8-40 房门示例

对于计算机网络中的“网关”，与前面的房门功能原理是类似的。网关与房门类似，它直接连接了两个网络，其作用是使直接连接的两个相邻网络间互通。如图 8-41 所示，网络 A 与网络 B 通过网关 1 直接连接了，所以网络 A 与网络 B 可以直接通过网关 1 实现三层互通；同理，因为网络 B 与网络 C 通过网关 2 直接连接了，所以网络 B 与网络 C 可以直接通过网关 2 实现三层互通。但是网络 A 不能直接通过网关 1 与网络 C 实现三层互通，同样网络 C 也不能直接通过网关 2 与网络 A 实现三层互通。

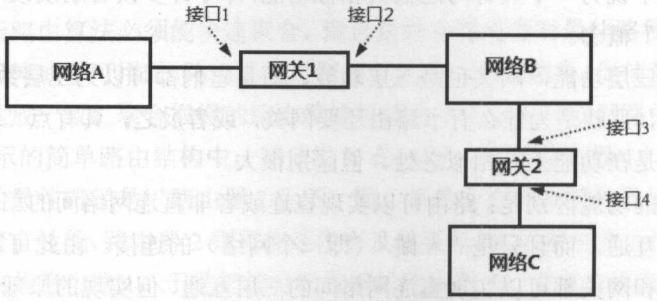


图 8-41　多个网关连接多个网络的示例

【说明】有专门的网关设备，这类设备通常没有路由功能，但更多的是同时具有网关功能和路由功能的设备，如三层交换机和路由器中的每个三层接口都可以作为所连网络的网关。在路由表项中的“下一跳”通常也是指网关，但也可以不是网关。

网关的两边各自通过一个网络接口（必须是三层的，配有 IP 地址）来连接各自的网络，就相当于房门是通过门的两面连接了两个房间一样。图 8-41 中的网络 A 到达网络 B 的网关是接口 1 的 IP 地址，网络 B 到达网络 A 的网关是接口 2 的 IP 地址，网络 B 到达网络 C 的网关是接口 3 的 IP 地址，网络 C 到达网络 B 的网关是接口 4 的 IP 地址。从这层意义上来说，直连网络间的网关配置是双向的、一组的。

在图 8-41 中，如果网络 B 中的有些用户想要通过网关配置到达网络 A，必须选择接口 2 的 IP 地址作为网关，此时网络 A 中的用户网关是接口 1 的 IP 地址；如果网络 B 中的另一些用户又想通过网关配置到达网络 C，必须选择接口 3 的 IP 地址作为网关，此时网络 C 中的用户网关是接口 4 的 IP 地址。

2. 多网关情形

说到这里，你或许又要问，一个网关设备上是否可以连接多个网络，作为多个网络的网关呢？当然可以，就像在一房间中可以安装多扇门，这样就可以从一个房间直接进入到多个房间。一台路由器或三层交换机上可能连接了多个网络，担当多个网络的网关。但是任意两个网络间的一组网关 IP 地址必须是不同的，也就是一组网关 IP 地址只能连接两个网络，供两个网络互访，就像一扇门的两面（相当于一组网关接口或者一组网关 IP 地址）只能连接两个房间一样。

这时你又肯定要问我了，既然一组网关 IP 地址只能实现两个网络的互通，那为什么一个三层

设备上连接的所有网络都可以直接访问呢？的确如此，这是因为这些网络都是直接连接在同一台三层设备上的，所以这些网络间都可看成是直接通过网关连接的，只不过不同网络间所使用的一组网关接口或者一组网关 IP 地址不同而已。

如图 8-42 所示，一个三层设备连接了网络 A、网络 B、网络 C 三个网络，它们之间是直接可以三层互通的。这时对于网络 A 访问网络 B 来说，一组网关接口就是接口 1 和接口 2，对于网络 A 访问网络 C 来说，一组网关接口就是接口 1 和接口 3，对于网络 B 访问网络 C 来说，一组网关接口就是接口 2 和接口 3。所以它们可以直接实现三层互通。

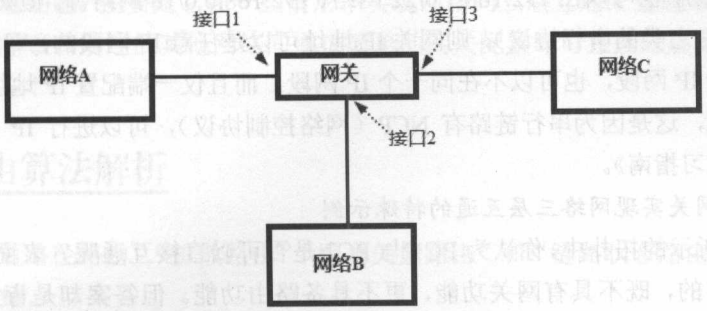


图 8-42　网关设备连接多个网络的示例

尽管对于任意一个网关接口都可以与其他任意网关接口形成一组，但这些网关接口组或者网关 IP 地址组中至少有一个是不一样的。如网络 A 中的网关接口 1，它可以与接口 2 或接口 3 形成一组，分别实现网络 A 与网络 B 或者网络 A 与网络 C 的互通。但不能通过其中任意一组来实现网络 A、网络 B、网络 C 三个网络的全部互通。

必须要说明的是，“网关”是需要在用户主机或者网络设备上具体配置才会生效的。因为这样一来，当有数据包中的目的 IP 地址在某个直连网络中时，就可以直接通过网关来转发这个数据包，而无需路由了。就相当于你在一个办公室问同事“总经理在哪里？”如果有人回答你说“在隔壁”，如果当前办公室只与一间房间直接相连，此时你还要问其他的吗？相信你一定会直接通过这唯一的房门进入到隔壁办公室找你的总经理了。但如果你当前的办公室直接连接了多个房间，那“隔壁”就有多个了，此时你肯定还会问是哪个“隔壁”，这时你同事肯定又会给你指明具体方向了，就相当于重新给你配置了新的网关，这样你就正确地找到通往正确“隔壁”的房门去找你的总经理了。

假设在图 8-42 中的各网络用户没有正确配置它们的网关，这时就不能直接通过网关来实现它们间的三层互通了。此时这 3 个网络是否仍可以直接三层互通呢？答案是肯定的，因为除了网关可以实现直接互连网络间的互访外，路由功能更是可以了，此时因为 3 个网络是直接连接在同一个设备上的，可以直接通过直连路由来实现这 3 个网络间的互通。

3. 网关与静态路由的区别

再来看一下网关与静态路由的区别。因为静态路由与网关更相像，它们都可以把数据包转发到

下一个网络，而且都只能转发一跳，要继续转发，必须再次依靠其他网关或者静态路由来实现。在静态路由配置中，我们通常要配置一个“下一跳 IP 地址”，这个 IP 地址通常就是指链路对端接口——网关接口的 IP 地址，但是也可以不是网关的 IP 地址，只要是通过其他路由可到达的即可。网关就不行了，只能是直接连接的接口 IP 地址。

网关 IP 地址或者下一跳 IP 地址根据不同网络类型有不同的要求：在以太网中，下一跳 IP 地址必须与本地网络用户 IP 地址在同一 IP 网段或者同在有包含关系的大的 IP 网段中（它们之间有包含关系），如本地网络用户的 IP 地址在 192.168.1.0/24，链路对端接口即网关接口 IP 地址可以在 192.168.1.10/24 网络，也可以在 192.168.1.0/22 网络、192.168.0.0/16 网络等。但如果是串行链路（运行 PPP 或者 PPPoE 之类的串行协议），则网关 IP 地址可以是任意 IP 网段的，即链路两端接口的 IP 地址可以在同一个 IP 网段，也可以不在同一个 IP 网段，而且仅一端配置 IP 地址，另一端的 IP 地址可以由对端分配，这是因为串行链路有 NCP（网络控制协议），可以进行 IP 地址协商。具体参见《华为路由器学习指南》。

4. 一个通过网关实现网络三层互通的特殊示例

在如图 8-43 所示的拓扑中，你认为 PC1 与 PC2 是否可以直接互通呢？表面看来是不可以的，因为交换机是二层的，既不具有网关功能，更不具备路由功能。但答案却是肯定的，此时我们只需要把二层交换机看成一条电缆，这样一来这个结构就相当于两个不同 IP 网段的主机直接相连。当然除此之外，还需要满足其他条件，那就是前面所说的在不同网络中网关 IP 地址的配置要求。

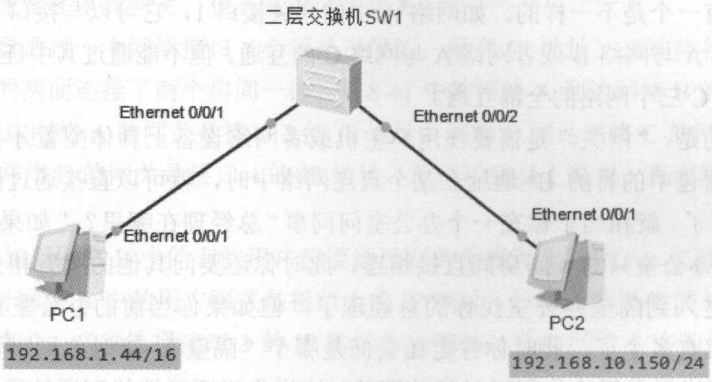


图 8-43　不同 IP 网段直接相连互通示例

本示例中各电缆均是双绞线，根据前面介绍的在以太网中网关的 IP 地址必须与本端接口 IP 地址在同一 IP 网段或者与本端接口 IP 地址所在网段有包含关系，而本示例 PC2 所在网络 192.168.10.0/24 包含在 192.168.0.0/16 网络中，所以是允许的。此时只需要把 PC1 的网关设为 192.168.10.150，PC2 的网关设为 192.168.1.44（也就是它们互为对方的网关）即可实现这两个主机直接的三层互通。但如果本示例中 PC1 和 PC2 所在网络不存在包含关系，则即使相互设对方为网关，也是不能直接三层互通的，大家不仿自己实验一下。

或许你会问，假设示例中的接口、电缆是串行的呢？本示例是不存在这种情形的，因为串行接口都是三层的，不可能在二层交换机上存在，而且主机的串口（COM）是不能配置 IP 地址的。如果示例中的交换机是三层交换机，并且 PC1 和 PC2 分别代表一个有配置了 IP 地址串口（Serial 口）的三层设备，前面已有介绍，在三层交换机（或路由器）上所连接的各个网络通过网关或者路由都可以直接实现三层互通，而且 PPP 链路两端接口的 IP 地址还可以不在同一 IP 网段。

最后总结一下：路由功能包括了网关功能，也就是说可以只有路由而没有网关，但不能只有网关而没有路由，因为网关只能对数据包进行一跳的转发，而路由可以进行多跳且不受限制的转发。之所以要有网关这项功能，主要是它在一些情形下实现网络间三层互通配置起来更为简单。如用户侧网络与直连的三层设备所连的其他网络间通过简单的网关配置即可直接三层互通，而不需要配置任何路由。

8.8 主要路由算法解析

本节要对“最短路径路由”“扩散路由”“距离矢量路由”和“链路状态路由”这 4 种主要的路由算法原理进行介绍。

8.8.1 最短路径路由算法

最短路径路由算法是一种从源结点一段段链路地向目的结点方向查找到达源结点链路长度最短的结点的方法，以此来确定一个结点到所有其他结点的最短路径。它的基本思想是构建通信子网拓扑图（不包括资源子网部分），每个结点代表一个路由器，这些路由器串接起来就代表一条通信线路。

在最短路径路由算法中，最典型的算法就是 Dijkstra 算法（迪克斯特拉算法），是由 Dijkstra 发明的。在该算法中，为了在一对给定的路由器结点之间选择一条最短路由路径，只需在通信子网拓扑图中找到在这起始和结束结点之间的中间结点串连起来后链路长度最短的路径（注意，这里是指线路长度）即可。它把最短路由的结点标识为工作结点，并且是永久性的结点，其到达源结点的距离值是不能改变的，其他的标识为临时性的结点，其到达源结点的距离可能会随工作结点的不同而改变。所有工作结点串联起来就是对应源结点和目的结点间的最短路由路径。

打个比方来说。现假设我们要从 A 城市到达 H 城市，如图 8-44 所示。我们肯定是先在 A 城市找一条与目的城市 H 同方向的最短出发路线（从图中可以看出共有 3 条路可选），假设由 A 到 B 这条路是最短的，所以先选择这条路线出发。

当我们到达 B 地点时，又要选择一条可以到达目的城市 H 的最佳路径（共有 2 条路线可选），这条路径要求离 B 地点最近，假设是选择到达 F 地点这条。然后在 F 地点继续寻找一条可以到达目的城市 H 的最短路线，因为 F 地点有一条路直接到达目的城市 H，所以直接选择这条路即可，最终到达了目的城市 H。

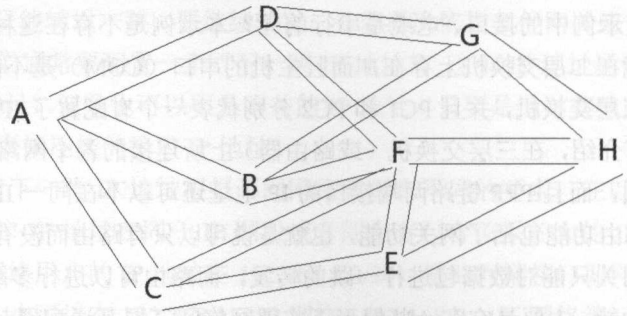


图 8-44　Dijkstra 算法比喻示例

以上就是 Dijkstra 最短路径算法的基本思想，下面通过一个具体的网络路由示例来加深理解。

如图 8-45 所示的子网图是一个典型的最短路径路由算法子网图，图中的每一个结点代表一台路由器，每条线段代表一条通信链路。现假设要使用 Dijkstra 算法计算结点 A 到结点 D 之间的最短路径。在网络中，路由器启动时需要先初始化，测量每条链路的长度，参见图中各条线段上的数字。下面是从 A 结点到达 D 结点的路由确定步骤。

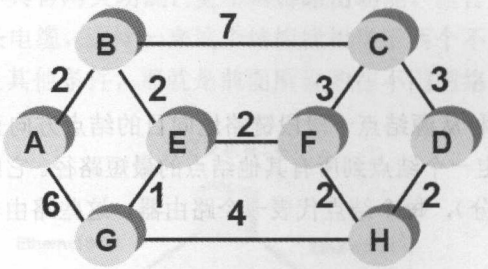


图 8-45　典型的 Dijkstra 算法子网图

（1）首先将源结点 A 标记为永久性工作结点（我们用箭头来特别标识），然后依次检查每一个与 A 结点直接连接的相邻结点，并且把它们与 A 结点之间的距离重新以(n,N)的方式进行标识，其中的 n 为与 A 结点相距的链路长度，N 为最近的工作结点。

因为本示例中与结点 A 直接相邻的结点只有 B 和 G，所以仅需标识这两个结点与 A 结点之间距离。此时的工作结点为 A，如图 8-46 所示，B 结点的标识为(2,A)，G 结点的标识为(6,A)，因为 B 结点到 A 结点的链路长度为 2，G 结点到 A 结点的链路长度为 6。其他与 A 结点不相邻的结点的距离标识为无穷远。

（2）比较 B 和 G 这两个结点与 A 结点之间的距离，可以看出 B 结点的距离更短，于是把 B 结点改为工作结点（箭头移到 B），同时变为永久性结点，其他结点（包括 G 结点）标注为临时结点。然后再以 B 结点为工作结点标记直接相邻的结点到源结点 A 的距离，当然对于前面已经计算过的结点将略过，如源结点 A 和 G 结点。

本示例中与 B 结点直接相邻的结点中，除了 A 结点外有 C、E 这两个结点。C 结点到达 A 结

点的距离就是 C 结点到 B 结点的链路长度 7，再加上 B 结点到 A 结点的链路长度 2，所以 C 结点到 A 结点的距离为 2+7=9，标识为(9,B)。同理，E 结点到 A 结点的距离为 2+2=4，标识为(4,B)，如图 8-47 所示。其他既不与 A 结点又不与 B 结点相邻的结点的距离仍标识为无穷远。

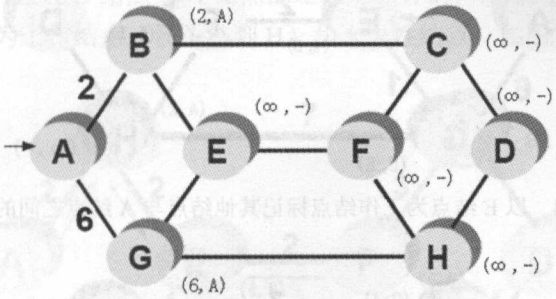


图 8-46　以 A 结点为工作结点标记其他相邻结点与 A 结点之间的距离

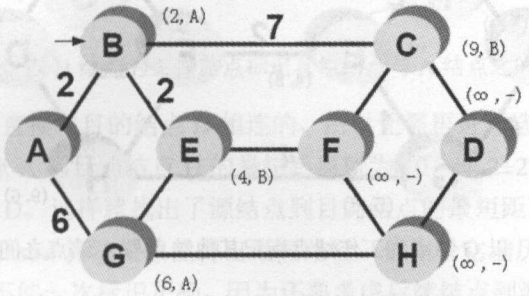


图 8-47　以 B 结点为工作结点标记其他结点与 A 结点之间的距离

（3）同样经过比较得出，E 结点到 A 结点之间的距离（为 4）比 C 结点到 A 结点的距离（为 9）近，所以此时把 E 结点改为工作结点（箭头移到 E），同时标注 E 结点为永久性结点，其他结点（包括 C 结点）标注为临时结点。

按同样的方法标记与 E 结点直接相邻的结点（包括结点 B、结点 G 和结点 F）到 E 结点的距离，但对于前面已经计算过的永久性结点 B 不再重新计算，而对于虽然原来已经计算过，但为临时性结点的 G 和 F 结点均需要重新计算。最终 G 结点标识改为(5,E)（在此步以前为(6,A)），F 结点标识为(6,E)，表示 G 结点和 F 结点到 A 结点的距离分别为 5 和 6，如图 8-48 所示。

（4）再用同样的方法比较 G 结点和 F 结点到 A 结点之间的距离，可以得出 G 结点更近，所以此时把 G 结点改为工作结点（箭头移到 G），同时标注 G 结点为永久性结点，其他结点（包括 F 结点）标注为临时结点，如图 8-49 所示。

再看一下与 G 结点直接相邻的结点，包括 A、E、H 这 3 个结点，但是 A、E 这两个结点在前面都已标识为永久性结点了，标识是不能更改的，所以在这里只需对 H 结点计算到 A 结点的距离。经过计算得出为(9,G)。

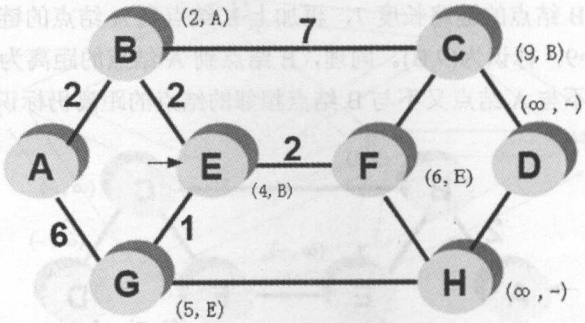


图 8-48　以 E 结点为工作结点标记其他结点与 A 结点之间的距离

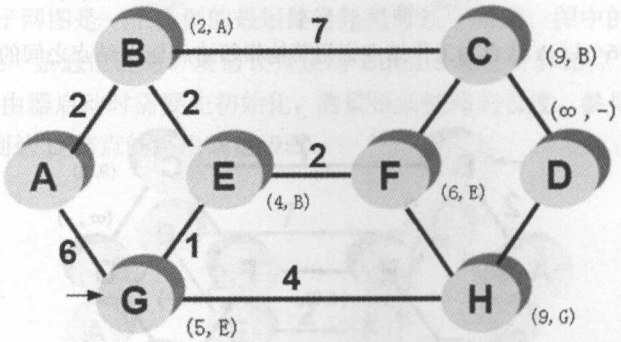


图 8-49　以 G 结点为工作结点标记其他结点与 A 结点之间的距离

在这里就要出现问题了，按照上面的计算，此时应该把 H 结点标识为下一个工作结点，但事实上，由 H 结点经 G 结点到 E 结点的距离（4+1=5）要长于由 H 结点经 F 结点到 E 结点的距离（2+2=4），所以经过后面的计算发现，在前面把 G 结点标识为永久性结点是错误的，这时要把 F 结点标识为工作结点（箭头移到 F），撤消 G 结点永久性工作结点的资格，如图 8-50 所示。

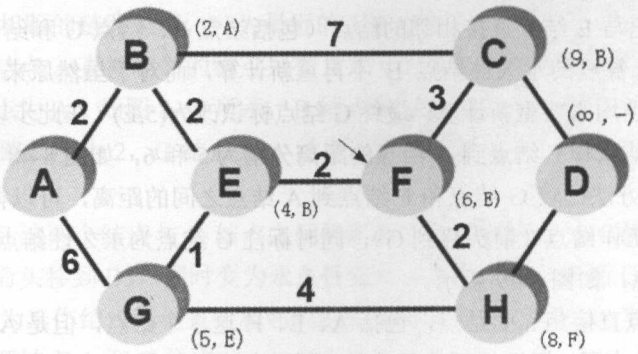


图 8-50　以 F 结点为工作结点标记其他结点与 A 结点之间的距离

（5）再检查与 F 结点直接相邻的结点，除了原来已标识为永久性结点的 E 外，其余就是 C 和 H 这两个临时结点了。重新计算它们到结点 A 间的距离，得到的值分别为 9 和 8。这里还要注意一个现象，就是对于 C 结点，本来属于临时结点，需要重新计算距离值，可是经过 F 结点到 A 结点的距离与原来计算所得的经过 B 结点到 A 结点的距离是一样的，所以距离值不需要改变。此时把距离较短的 H 结点标识为工作结点（箭头移到 H）和永久性结点，如图 8-51 所示。

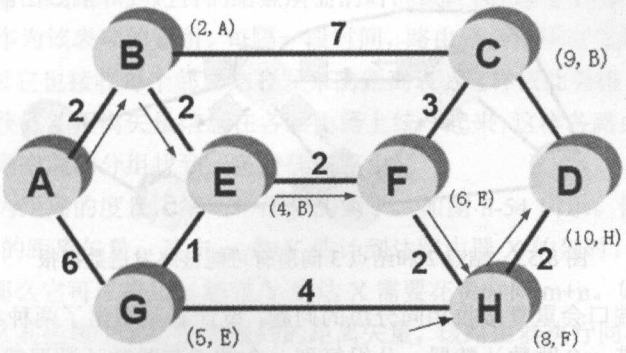


图 8-51　以 H 结点为工作结点标记其他结点与 A 结点之间的距离

此时，因为 H 结点是直接与目的结点 D 相连的，所以无需再进行选举了，直接标识 D 结点的距离为(10,H)。即从源结点 A 到目的结点 D 的最短距离就为 10，即 2+2+2+2+2，如图 8-51 所示的连线：A→B→E→F→H→D。这样就找出了源结点到目的结点的最短距离。

从以上可以看出，Dijkstra 算法虽然能得出最短路径，但由于它遍历计算的结点很多，所以效率低。另外，有些结点还不能一次标识正确，因为还要考虑后续结点到源结点的距离，如以上示例中 G 结点和 F 结点的工作结点标识，最初的标识就是错误的，因为它没有考虑后续结点到源结点的距离。

8.8.2　扩散路由算法

扩散（Flooding）路由算法最初原始的方法是把每一个接收到的分组从除接收端口之外的所有其他端口上转发出去。如图 8-52 所示是数据分组在源结点 1 上分散到了 3 个链路。

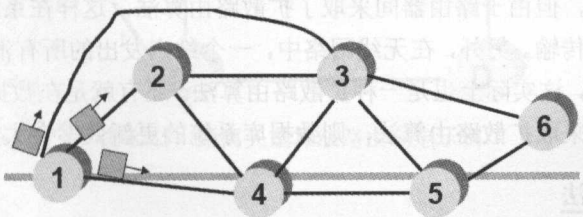


图 8-52　结点 1 分散成 3 个链路发送数据报

很显然，原始的扩散路由算法会产生大量的分组副本，如在图 8-53 中，当数据分组到了结点

2 和结点 3 后都会向所有可能的链路上发送分组，这样就会出现分组重复。而且越后面的结点收到的重复数据分组越多。如图中的结点 3，它既有直接从结点 1 发来的分组，又有经过结点 2 转发的同一分组，还有从结点 4 发来的同一分组。不仅如此，这种扩散法还会造成分组死循环，如结点 3 发来的分组，它又会向结点 2 返回一样的分组。这样，即使是真正的目的结点，也可能会接收到多个同样的分组，这显然是不必要的和没有任何意义的。

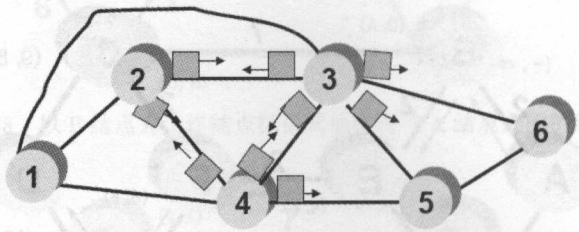


图 8-53　结点 2 和结点 3 向所有可能链路发送数据报

为了解决以上各端口会重复接收相同分组的问题，设计者又想出了两种办法。其中一种办法是在每个分组头部中携带一个跳数计数器，分组每到一个结点其跳数计数器就减 1，当计数器减到 0 时该分组将被丢弃。而计数器的初始值可以设为通信子网的直径，即相距最远的两个结点之间的跳数，这样可以减少那些循环转发的分组。

另一种办法是在结点上记住哪些分组已经转发过了，对于重复接收的分组直接丢弃，从而确保一个分组不会被同一个结点转发两次。这要求源结点从主机收到一个分组后为分组分配一个序号，同一分组的序号相同，就像我们在第 6 章介绍数据帧的发送时为每个帧分配一个序号一样，重发的同一数据帧序号相同，这样接收端就可以确保对同一个帧只接收一次。同时每一个结点对于每一个源结点都要维护一张序号表，记录收到的来自同一源结点的所有分组序号。每当一个结点收到来自某个源结点的分组时，就用分组的序号去查找该源结点的序号表，如果该序号已在表中则该分组被丢弃。为了防止序号表过大，序号表中还应增设一个计数器 k，表示序号直至 k 的分组都已经转发过了，从而不需要保留序号小于 k 的序号。

尽管扩散路由算法有它的不足，但也不是说它一无用处。由于它的数据分组的分散传送，使它的强壮性非常好，在一些经常可能遇到破坏的网络系统中是非常需要的。如军事网络系统，可能在战争中被瞬间炸得粉碎，但由于路由器间采取了扩散路由算法，这样在系统中只要存在一个链路，数据就仍能完整地向前传输。另外，在无线网络中，一个站点发出的所有消息可以被位于无限距离范围内的所有站点接收，这实际上也是一种扩散路由算法。还有就是在数据库系统中，如果数据库系统中有多台服务器，采取扩散路由算法，则数据库系统的更新效率会大大提高。

8.8.3　距离矢量路由算法

现代计算机网络通常使用动态路由算法，因为这类算法能够适应网络拓扑和流量的变化，其中最流行的两种动态路由算法是“距离矢量路由算法”和“链路状态路由算法”。

距离矢量（Distance Vector，DV）路由算法是 ARPANET 网络上最早使用的路由算法，也称 Bellman-Ford 路由算法和 Ford-Fulkerson 算法，主要在 RIP（Route Information Protocol）协议中使用。Cisco 的 IGRP 和 EIGRP 路由协议也是采用 DV 这种路由算法的。

距离矢量路由算法的基本思想为：每个路由器维护一个距离矢量（通常是以延时作变量的）表，然后通过相邻路由器之间的距离矢量通告进行距离矢量表的更新。每个距离矢量表项包括两部分：到达目的结点的最佳输出线路和到达目的结点所需的时间或距离，通信子网中的其他每个路由器在表中占据一个表项并作为该表项的索引。每隔一段时间，路由器会向所有邻居结点发送它到每个目的结点的距离表，同时它也接收每个邻居结点发来的距离表。这样依此类推，经过一段时间后便可将网络中各路由器所获得的距离矢量信息在各路由器上统一起来，这样各路由器只需要查看这个距离矢量表就可以为不同来源的分组找到一条最佳的路由。

现假定用延时作为距离的度量，举一个简单的例子，如图 8-54 所示。假设某个时候路由器 Y 收到其邻居路由器 X 的距离矢量，其中 m 是 Y 估计到达路由器 X 的延时。若 Y 路由器知道它到邻居 Z 的延时为 n，那么它可以得知 Z 通过 Y 到达 X 需要花费时间 m+n。如果 Z 路由器还有其他相邻路由器，则对于从其他每个邻居那里收到的距离矢量，该路由器执行同样的计算，最后从中选择费时最小的路由作为 Z 去往 X 的最佳路由，然后更新其路由表并通告给其邻居路由器。

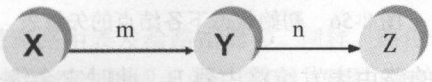


图 8-54　距离矢量路由算法简单实例

现以一个如图 8-55 所示的示例介绍距离矢量路由算法中的路由的确定流程，各段链路的延时均已在图中标注。A、B、C、D、E 代表 5 个路由器，假设路由表的传递方向为：A→B→C→D→E（这与路由器启动的先后次序有关）。

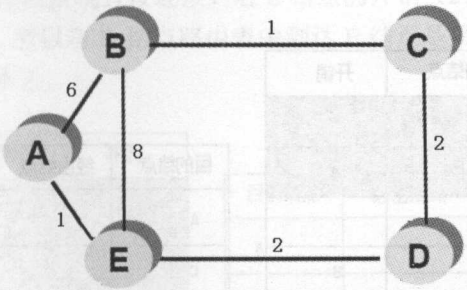


图 8-55　距离矢量算法路由确定示例

具体流程如下：

（1）初始状态下，各路由器都只收集直接相连的链路的延时信息，各路由器结点得出各自的初始矢量表如图 8-56 所示。因为各结点间还没有交换路由信息，所以它们的初始状态的路由表也如它们的矢量表。

A结点初始矢量表

源 \ 目的	B	C	D	E
A	6			1

B结点初始矢量表

源 \ 目的	A	C	D	E
B	6	1		8

C结点初始矢量表

源 \ 目的	A	B	D	E
C		1	2	

D结点初始矢量表

源 \ 目的	A	B	C	E
D			2	2

E结点初始矢量表

源 \ 目的	A	B	C	D
E	1	8		

图 8-56　初始状态下各结点的矢量表

（2）现在路由器 A 把它的路由表发给路由器 B。此时它会综合从 A 路由器发来的路由表和它自己的初始路由表更新为一个新的矢量表，如图 8-57 左图所示（最终的矢量表如图中深颜色部分所示）。从图中可以看出，从 B 结点到达 E 结点此时存在两条路径，一条是直达的，一条是通过 A 结点到达的。而且这两条线的开销不同，经过 A 结点到达 E 结点的开销（7）比直达线路的开销（8）更低，所以最终在形成的路由表中把到达 E 结点的线路改为经由 A 结点这条线路，如图 8-57 右图所示。

目的结点	经由结点	开销
A		6
C		1
D		
E		8
E	A	7

目的结点	经由结点	矢量开销
A		6
C		1
E	A	7

图 8-57　B 结点新的矢量表和路由表

（3）B 再把最终形成的路由表发给路由器 C。同样，路由器 C 也要把它原来的初始路由表与从 B 路由器发来的路由表进行综合形成新的矢量表，如图 8-58 左图所示（最终的矢量表如图中深

颜色部分所示)。在新的矢量表中，除了最初的直接连接的 B 和 D 结点间的矢量外，还新收集了到达 A 和 E 结点的矢量信息。因为 C 结点没有与 A 和 E 结点的直接连接，在初始路由表中并没有到达这两个结点的路由信息，所以现在只有采用从路由器 B 发来的路由表中经过 B 结点到达 A、E 结点的路径。

这里需要注意一点，因为在 B 结点路由表中就已识别了直接通过 B 结点到达 E 结点的开销(8)还比依次通过 B、A 结点到达 E 结点的开销（7）大，所以在 C 结点路由表中是采用依次通过 B、A 结点到达 E 结点这条路径。最终形成的路由表如图 8-58 右图所示。

目的结点	经由结点	开销
A	B	7
B		1
D		2
E	B→A	8

目的结点	经由结点	矢量开销
A		7
C		1
D		2
E	B→A	8

图 8-58　C 结点新的矢量表和路由表

（4）路由器 C 再把它的最终路由表发给路由器 D。同样，路由器 D 也要把它原来的初始路由表与从路由器 C 发来的路由表进行综合形成新的矢量表，如图 8-59 左图所示（最终的矢量表如图中深颜色部分所示)。在新的矢量表中，除了最初的直接连接的 C 和 E 结点间的矢量信息外，还新收集了到达 A 和 B 结点的矢量信息。因为 D 结点没有与 A 和 B 结点的直接连接，所以在其最初的路由表中并没有到达这两个结点的矢量信息，此时仍采用经过 C 结点到达 A 和 B 结点的路径。

在这里同样需要注意一点，从 D 结点到达 E 结点也有两条路径：一是直接到达，二是依次通过 C、B、A 结点到达，经过比较发现直接连接到达 E 结点的开销（2）要比通过 C、B、A 结点到达 E 结点的开销（10）要小，所以在 D 结点路由表中到达 E 结点是采用直接连接这条线路。最终形成的路由表如图 8-59 右图所示。

目的结点	经由结点	开销
A	C	9
B	C	3
C		2
E	C→B→A	10
E		2

目的结点	经由结点	矢量开销
A	C	9
B	C	3
C		2
E		2

图 8-59　D 结点新的矢量表和路由表

（5）路由器 D 再把它的最终路由表发给路由器 E。同样，路由器 E 也要把它原来的初始路由表与从路由器 D 发来的路由表进行综合形成新的矢量表，如图 8-60 左图所示（最终的矢量表如图中深颜色部分所示）。在新的矢量表中，除了最初的直接连接的 A、B 和 D 结点间的矢量外，还新收集了到达 C 结点的矢量信息，因为 E 结点没有与 C 结点的直接连接。此时仍采用经过 D 结点到达 C 结点的路径。

在这里有两个需要注意的地方：一是从 E 结点到达 A 结点的路径问题，因为此时 E 结点与 A 结点是直接连接的，而且其开销（1）要比原来从路由器 D 发来的路由表中提供的通过 D、C、B 结点到达 A 结点的开销（11）要小，所以在最终的 E 结点路由表中到达 A 结点是采用直接连接这条线路；二是 E 结点虽然也是与 B 结点直接连接，但它的开销（8）还要比原来从路由器 D 发来的路由表中提供的依次经过 D、C 这两个结点到达 B 结点的开销（5）大，所以在最终的 E 结点路由表中到达 B 结点是采用依次经过 D、C 两个结点这条路径。最终形成的路由表如图 8-60 右图所示。

目的结点	经由结点	开销
A		1
A	D→C→B	11
B		8
B	D→C	5
C	D	4
D		2

目的结点	经由结点	矢量开销
A		1
B	D→C	5
C	D	4
D		2

图 8-60　E 结点新的矢量表和路由表

通过以上步骤，网络中各路由器就完成了整个路由表的确定，当然在拓扑结构发生变化时，各路由器的路由表又会发生变化，重新进行更新。

8.8.4　链路状态路由算法

上节介绍的距离矢量路由算法简明扼要，这是它在早期网络几乎无规模可言时能得到广泛应用的主要原因。在网络拓扑结构相对简单且链路极少发生故障时，这种算法的效果完全可以令人满意；然而对于庞大而复杂的网络，该算法计算新路由的收敛速度极慢，而且在它进行计算的过程中网络处于一种过渡状态，极可能发生循环并造成暂时的拥塞。再者，当前网络底层链路技术多种多样，带宽各不相同，而距离矢量路由算法对此则视而不见。链路状态路由算法就是为了克服这些缺点才提出的，代表协议如 OSPF、IS-IS 等。

链路状态路由（Link State Routing）算法的基本思想是：网络中各个节点不必交换通往目的站点的距离，而是维护一张网络拓扑图，在网络拓扑结构发生变化时及时更新拓扑图即可。隐藏在链

路状态路由算法之后的思想十分简明，它由以下 5 个步骤组成：

（1）发现邻居结点并知道其网络地址。

（2）测量到各邻居结点的延时或开销。

（3）组装一个分组以告之刚知道的所有信息。

（4）将这个分组发送给所有其他路由器。

（5）使用 Dijkstra 算法（即“最短路径”算法）计算到每个其他路由器的最短路径。

实际上，运行链路状态算法并非一定要采用 Dijkstra 算法，唯一必须遵从的要求是所有结点都应该使用完全相同的度量制式，也就是说，不论使用什么算法，只要能找出相同的最短路径即可。下面简单介绍一下以上 5 个步骤。

1. 发现邻居

当一个路由器启动时，它的第一个任务就是找出哪些路由器是它的邻居。为了实现这个目标，它只需要在每一条点到点线路上发送一个特殊的 Hello 分组即可。线路另一端的路由器应发送一个应答来说明它是谁，就像我们打电话一样，当一方发出问候时，对方会作出回应，以证明对方就是打电话一方要找的人。路由器的这些名字必须是全局唯一的。

如图 8-61 所示就是一个由 A 结点向其邻居结点发送 Hello 请求分组 H_Re，然后邻居结点在收到其 Hello 分组后发回 Hello 应答分组 H_Rp 的示意图。

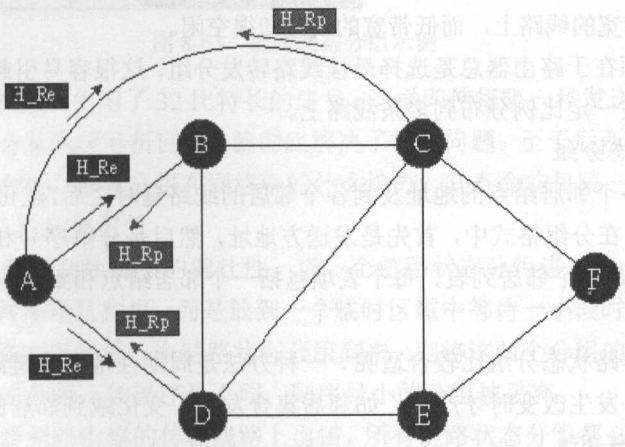


图 8-61　发送 Hello 请求分组和返回 Hello 应答分组的示意图

2. 测量线路延时

测量线路延时最直接的方法是路由器在线路上发送一个特殊的 ECHO 分组，收到 ECHO 分组的路由器必须立即发回一个应答，通过测量发送与接收之间的时间（来回时间）并除以 2 可以得到单向的线路延时。可以多测几组数据，求平均线路延时，然后即可得知各段链路之间的延时，亦即开销（Cost），如图 8-62 所示。

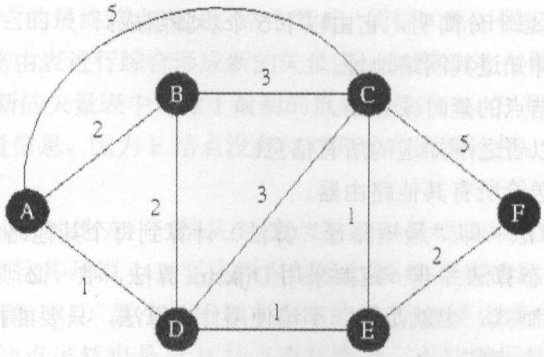


图 8-62　计算链路开销示意图

显然以上我们假设延时是对称的，也就是分组发送和返回的时间是一样的，但实际上往往并不总是这样。在计算线路延时时要不要考虑线路上的负载，这是一个颇有争议的问题。如果考虑线路上的负载，那么发送时间应从 ECHO 分组放入队列时算起；如果不考虑线路负载，那么发送时间应从 ECHO 分组到达队头时算起。考虑线路负载意味着当两条线路的带宽相同时，路由器将选择负载较轻的线路作为最佳输出线路，这种选择能导致良好的性能。但这会使轻负载的线路很快超载，从而另一条线路又成为最佳选择，这样容易引起路由表的剧烈振荡，导致不稳定路由和其他潜在问题的产生。只考虑带宽不考虑负载就不会出现这种问题，但完全不考虑负载也是不合理的，因为它会使流量都集中到高带宽的线路上，而低带宽的线路却很空闲。

以上问题的根源在于路由器总是选择最佳线路转发分组，这很容易引起负载不均衡，因此较明智的做法是将负载按一定比例分布到多条线路上。

3. 构造链路状态分组

一旦收集到了各个邻居结点的地址及到各个邻居的线路延时之后，就可以构造一个链路状态分组来携带这些信息。在分组格式中，首先是发送方地址，然后是分组序号和时间（亦即链路状态生存时间 TTL），最后是一个邻居列表，每个表项包括一个邻居结点和到达这个邻居结点的总延时，如图 8-63 所示。

什么时候构造链路状态分组比较合适呢？一种办法是周期性地产生链路状态分组；另一种办法是仅在发现网络拓扑发生改变时才产生，如邻居集合发生了变化或到邻居的线路延时发生了变化。

4. 发送链路状态分组

该算法最复杂的部分是如何可靠地发送链路状态分组。当分组被发送和安装后，先得到分组的路由器就会用它来改变自己的路由表，此时不同的路由器可能会使用不同的网络拓扑版本，这可能导致不一致、路由环路、不可到达结点和其他问题。

为了可靠地发送链路状态分组，采用了扩散算法（这在前面已有介绍，参见 8.8.2 节）。每个分组携带一个序号，每当发送一个新的分组时，分组序号就加 1。每个路由器维护一张由源路由器、分组序号组成的列表，记录从每个源路由器收到的最新的分组序号。当一个链路状态分组到来时，用分组中的源路由器地址和序号检查列表，如果这是一个新的链路状态分组，就用扩散算法再将其转

发出去，否则将分组丢弃。采用序号来区分新旧分组是会有一些问题的：一是当序号折回（因为序号也是有限制的，毕竟序号所占的比特位是限定的）时，新的分组都会被丢掉，因为它们与原来接收的分组号一样；二是当路由器发生崩溃并重启后，只能从序号 0 开始发送，这些分组也会被丢弃；三是当序号在传输过程中出错时，如 4 变成了 65540，则其后大量的分组都会被丢掉，因为此时路由器认为对应源路由器上一个最近的分组号就是 65540，后面的分组号必须大于这个值，否则就认为是重复接收的。



图 8-63 链路状态分组示例

为解决序号折回的问题，使用了 32 比特长的序号，这样即使每隔一秒发送一个链路状态分组，也需要 137 年的时间才会发生序号折回，也就彻底解决了折回问题。至于后两个问题，解决的办法是在分组中增加一个寿命域，不管分组在网络中的什么地方，其寿命均每隔一秒减 1，当寿命减为 0 时分组被丢弃。这种方法显然很笨。

还有一些改进的措施可增加算法的健壮性。当一个链路状态分组进入一个路由器并需要扩散时，该分组并不马上放到传输队列里，而是放到一个临时区域中等待一小段时间。如果在该分组被发送前，来自同一个源路由器的另一个链路状态分组到来，则将这两个分组的序号进行比较。如果序号相同，就丢弃后来的分组；如果序号不同，则序号小的分组被丢弃。

为防止分组在路由器到路由器的传输线路上出错，所有链路状态分组都必须被确认。当线路空闲时，临时区域被循环扫描，从中选择一个分组或确认进行发送。路由器将收到的链路状态分组存放在分组缓冲器中。每一行对应一个最近收到的但还未处理完毕的链路状态分组，除记录该分组的源路由器、序号、生存时间、数据以外，对应每一条输出线路还有发送和确认两个标志位，用来指示是否要在这条线路上转发或确认这个分组。

5. 计算新路由

一旦一个路由器收集齐了一整套链路状态分组，它就可以建立完整的通信子网图。在这张图中每条链路均被报告了两次（因为一条链路连接了两个结点，它们会分别进行计算的），这两个值可

以分开用，也可以求平均值。通信子网图建立起来后，就可以运行 Dijkstra 算法求路由器到任何目的路由器的最短通路。如图 8-64 所示就是图 8-62 所示网络的最终路由表（各路由器上一样）。

路由器A		路由器B		路由器C		路由器D		路由器E		路由器F	
目的地	下一站	目的地	下一站	目的地	下一站	目的地	下一站	目的地	下一站	目的地	下一站
A		A	A	A	D	A	A	A	D	A	E
B	B	B		B	B	B	B	B	D	B	E
C	D	C	C	C		C	E	C	C	C	E
D	D	D	D	D	E	D		D	D	D	E
E	D	E	D	E	E	E	E	E		E	E
F	D	F	D	F	E	F	E	F	F	F	

图 8-64　最终的链路状态路由表示例

8.9　网络拥塞控制方法和原理

网络拥塞现象是指到达通信子网中某一部分的数据报数量过多，使得该部分网络来不及处理，以致引起这部分乃至整个网络性能下降的现象。当拥塞严重时，甚至会导致网络通信业务陷入停顿，即出现死锁现象。这种现象与公路网中经常所见的交通拥挤一样，在上班高峰时间段中车辆大量增加，在有限的道路宽度下，大量车辆的涌入使各车辆的行驶速度大受限制，也就出现拥塞现象，这样每辆车到达目的地的时间都相对增加（即延迟增加），甚至有时还会因拥塞而长时间无法开动，这就是出现了局部死锁现象。

之所以会出现拥塞现象，就是因为通信子网中所承受的负荷（即通信子网中正在传输的数据报数）超出了网络的吞吐能力（包数/秒）。当通信子网负荷比较小时，网络的吞吐量随网络负荷的增加而线性增加。当网络负荷增加到某一值后，此时网络的吞吐量达到了最大值。若再增加网络负荷，此时的网络吞吐量会不升反降，表明网络中开始出现拥塞现象了。

就像路上的车一样，当路上车辆数适当时，车辆可以正常行驶；但当路上的车辆数再次增加时，路上车辆的行驶速度会慢下来，此时道路的吞吐量会降下来（此时的“吞吐量”指每秒通过的车辆数）。在一个出现拥塞现象的网络中，到达某个结点的数据报将会遇到无缓冲区可用的情况，出现丢弃现象，从而使这些数据报不得不由源结点重传，从而使通信子网的有效吞吐量下降。由此引起恶性循环，使通信子网的局部甚至全部处于死锁状态，最终导致网络有效吞吐量接近为零。

8.9.1　网络拥塞控制方法

为了确保网络畅通，保持网络的有效吞吐量，就必须对网络中的拥塞现象进行有效控制，同时

要尽量避免死锁现象的发生，这就是本节所要介绍的“拥塞控制”（Congestion Control）方法。但“拥塞控制”与“流量控制”（Flow Control）不是完全等同的，“拥塞控制”需要确保通信子网能够承载用户提交的通信量，是一个全局性问题，涉及主机、路由器，以及与降低网络传输性能有关的所有因素等很多因素；而“流量控制”与点到点的通信量有关，主要解决快速发送方与慢速接收方的问题，是局部问题，控制的方法是抑制发送端发送数据的速率，以便使接收端来得及接收或处理，一般都是基于反馈进行控制的，具体可以参见第 6 章数据链路层中介绍的流量控制方法。

产生拥塞的原因主要有 3 个方面：结点的缓存区容量太小、输出链路的信道带宽不够、处理机的速度不够快。目前通常采用的网络拥塞控制方法有 3 种：缓冲区预分配法、分组丢弃法和定额控制法。

1. 缓冲区预分配法

“缓冲区预分配”（Buffer Preallocation）拥塞控制方法常用于虚电路分组交换网中。在虚电路建立时，要求呼叫请求分组所途经的每个结点为此条虚电路预先分配一个或多个数据缓冲区。若某个结点当前没有可用的缓冲区（被请求的结点会返回一个“忙”应答信号给呼叫请求分组发送者），则在建立虚电路时让呼叫请求分组选择其他路径来建立虚电路。这样，通过途经的各个结点就可以为每条虚电路开设永久性的缓冲区（直到虚电路拆除），就总能有空间来接纳并转送经过的数据报。当结点收到一个数据报并将它转发出去之后，该结点向发送结点返回一个确认信息。该确认一方面表示接收结点已正确收到数据报，另一方面告诉发送结点，该结点已空出缓冲区以备接收下一个数据报。

打个比方来说。假设有一批货物要从甲地点运输到乙地点，沿途有 A、B、C 三个中转站点（相当于虚电路途经的中间结点）可选，如图 8-65 所示。因为货物较多，为了提高货运周转率，除了要求中转站点有转强的转运能力外，还需要中转站点有较大的仓储空间（相当于用于解决网络拥塞的缓冲区）可用，以便把那些中转站点来不及转运的货物先保存在仓库中，否则就可能有些来不及转运的货无地可放了。

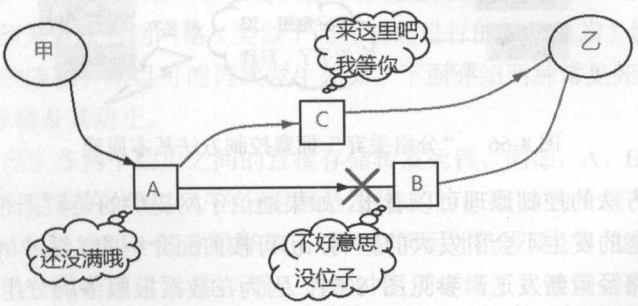


图 8-65 缓冲区预分配法

本来打算走“甲－A－B－乙”这条运输路线，结果在发货方甲电话了解后发现 B 站点没有周转用的仓库可用了（相当于中间结点没有缓冲区可用），于是发货方甲打电话问可选的 C 站点，被

告知有仓库可用，于是最终选择了甲－A－C－乙这条货运路径。相当于在建立虚电路时另选其他路径。

当货物正常运输时，每个中转站点都会向发货方甲告知货物转运情况（相当于数据传输时接收方发送的确认分组），这时发货方一方面可以得知哪些货物正常转运了，另一方面也可估算出对应中转站点仓库空间的余量，可以更好地安排发货进度。

以上只是一种同一时刻允许单一数据报处理的情况，若结点之间的协议允许多个未处理的数据报存在，则为了完全消除拥塞的可能性，每个结点要为每条虚电路保留等于窗口大小数量的缓冲区（相当于图 8-65 所示的示例中要求中转站点有可以保存多少吨货物的可用仓库空间），而且不管实际上有没有通信量（也就是相当于图 8-65 中各中转站点的仓库是否可以存满货物）。这样一来，在实际的网络通信中会有非常多的资源（线路容量或存储空间）被某个连接占有，因此网络资源的有效利用率较低。所以，这种拥塞控制方法主要用于要求高带宽和低延迟的场合，例如传送数字化语音信息的虚电路，是以牺牲资源来换取性能保证的。

2. 分组丢弃法

“分组丢弃”（Packet Dropping）拥塞控制方法是不必预先为数据报预留缓冲区，而是在路由器的缓冲区已被占满时只需将后面到来的数据报丢弃，仅在缓冲区腾出空间后再重新接收数据分组。

如图 8-66 所示，从源站点到目的站点间经过两个路由器，当某个时间路由器 2 发现自己的缓冲区满了，不能再接收由路由器 1 转发来的分组时，就会把后来发来的分组全部丢弃，如图中的 packet-n 及以后序号的分组。在虚电路服务中，通过计时器控制方法可以使发送被丢弃分组的结点重发这些分组，如图中路由器 1 重发的 packet-n 分组。很显然，这是一种最直接同时也最粗的拥塞控制方法，因为是直接拒收后面到来的数据报。

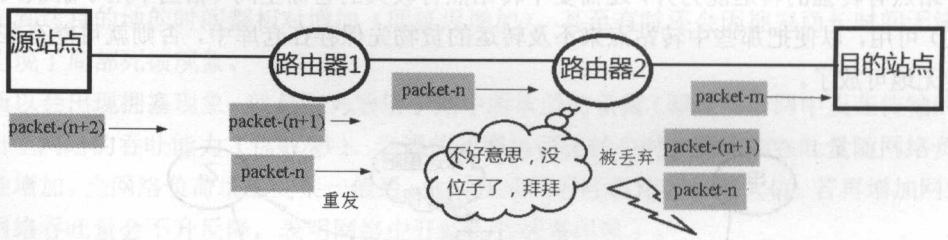


图 8-66 “分组丢弃”拥塞控制方法基本原理

从这种拥塞控制方法的控制原理可以看出，如果通信子网提供的是数据报服务，则采用这种拥塞控制方法来防止拥塞的发生不会引发大的影响，仍可按前面介绍的“缓冲区预分配”拥塞控制方法来解决，选择其他路径重新发送，参见图 8-63。因为在数据报服务的分组报文中每个分组都有完整的路由地址信息，可以独立地选择最佳路径进行路由，当发现原来的路径性能不佳时，会自动重新以其他更好的路径进行分组交换；另外，在数据报服务中不需要接收结点在接收到分组后进行确认，所以即使接收结点把数据报丢弃了，也不会引起发送结点长时间等待确认应答。

如果通信子网提供的是虚电路服务，则必须在某个前面的结点中保存了被丢弃数据报的备份，

以便拥塞解决后能重新传送。因为虚电路服务的分组报文中没有独立的路由地址信息，也不能单独地被路由，同一数据报的所有分组必须走同一条虚电路，而且必须要求接收结点在收到每个分组后有确认应答。

如果虚电路中某个数据报的某个分组报文被丢弃了，发送结点就不会收到对应分组的应答确认。这里又有两种解决被丢弃数据报重发的方法：一种方法是采用计时器方法，当某个分组发送后，在计时器超时后仍没有收到接收结点发回的对应分组确认应答，就要使某个转发该数据分组的结点重新发送对应的数据分组，直至数据分组被收到，但每一结点的缓冲区可用空间是在不断动态变化的，因为它们在接收数据分组的同时也在不断进行转发数据分组；另一种方法是让发送被丢弃数据报的结点在尝试重发一定次数后放弃发送，并迫使数据源结点超时而重新开始发送。

3. 定额控制法

“定额控制”拥塞控制方法是在通信子网中设置适当数量，即称为“许可证”的特殊信息来实现的。其中一部分许可证在通信子网开始工作前预先以某种策略分配给各个源结点，另一部分则在子网开始工作后在网中四处环游。当源结点要发送来自源端系统的数据报时，它必须首先拥有许可证，并且每发送一个数据报注销一张许可证。目的结点方则每收到一个数据报并将其递交给目的端系统后，便生成一张许可证。这样便可确保子网中的数据报数量不会超过许可证的数量，从而防止拥塞的发生。但这时网络的传输效率可能不能发挥到最佳。

其实这种控制方法有点像令牌环网上的介质争用解决方案。在令牌环网中，源结点要发送数据必须要获取网络中唯一的令牌，不同的是这里的证书不是唯一的，可以有许多。

8.9.2 死锁及其预防

“死锁”就是一些结点由于没有空闲缓冲区而无法接收和转发数据报（转发时也需要一定的缓冲空间），这样的结果是结点间相互等待，都既不能接收数据报也不能转发数据报，并一直保持这一僵局，严重时甚至导致整个网络瘫痪。

“死锁”是“拥塞”的极端后果，是网络中最容易发生的故障之一，即使在网络负荷不很重时。此时，只能靠人工干预来重新启动网络（类似于因死机而进行的系统重启）解除死锁，但重新启动后并未消除引起死锁的隐患，所以可能再次发生死锁。下面介绍两种常见死锁情况及预防方法。

（1）存储转发死锁及其防止。

最常见的死锁是发生在两个结点之间的直接存储转发死锁。例如，A、B 和 C 三个结点的缓冲区都满了，不能再接收更多的分组。A 的分组都是发往 B 的，而 B 由于缓冲区满了不能接收任何分组。这样，A 要等到 B 发送掉一些分组释放出缓冲空间后才能发送。B 的分组是送往 C 的，而 C 的缓冲区也满了。C 的分组是送往 A 的，而 A 的缓冲区同样满了。换句话说，A 等待 B 清空缓冲区，B 等待 C 清空缓冲区，而 C 则等待 A 清空缓冲区，但事实上当前情况下这 3 个结点的缓冲区都已无法清空了。当一个结点处于死锁状态时，所有与之相连的链路将完全被拥塞。

一种防止存储处于死锁的方法是为每个结点设置 m+1 个缓冲区（m 为通信子网的直径，即从任一源结点到任一目的结点间的最大链路段数）并顺序编号。每个源结点仅当其 0 号缓冲区可

用时才能接收源端系统来的数据报，而此数据报仅能转发给 1 号缓冲区空闲的相邻结点，再由该结点将数据报转发给它的 2 号缓冲区空闲的相邻结点，依此类推。最后，该数据报或顺利到达目的结点并被递交给目的端系统；或到了某个结点编号为 m 的缓冲区中再也转发不下去，此时一定发生了循环，应该将该数据报丢弃。由于每个数据报都是按照编号递增规则分配缓冲区的，所以结点之间不会相互等待空闲缓冲区而发生死锁现象。这种方法的不足之处在于当某个结点虽然有空闲缓冲区，但正巧没有所需要的特定编号的缓冲区时，数据报仍要等待，从而造成了缓冲区和链路的浪费。

另一种防止存储转发死锁的方法是使每个数据报上都携带一个全局性的惟一的“时间戳”，每个结点要为每条链路保留一个特殊的接收缓冲区，而其他缓冲区均可用于存放中转数据报，按每条输出链路队列上的数据报时间戳顺序排队。例如，A 结点要将数据报送到 B 结点，若 B 结点没有空闲缓冲区，但正巧有要送到 A 结点的数据报，此时 A、B 结点可通过特殊的接收缓冲区交换数据报；若 B 结点既没有空闲缓冲区，也没有要送到 A 结点的数据报，B 结点只好强行将一个出路方向大致与 A 结点方向相同的数据报与 A 结点数据报互相交换，但此时 A 结点中的数据报必须比 B 结点中的数据报具有更早的时间戳，这样才能保证子网中某个最早的数据报不受阻挡地转发到目的地。由此可见，每个数据报最终总会一步一步地发送到目的结点，以此来避免死锁现象的发生。

（2）重装死锁及其防止。

“重装死锁”就是在数据分组重新组装、恢复成原来数据报的过程中发生的死锁，是另一种比较严重的死锁现象。假设源结点发送的报文很长，被源结点拆成若干个数据分组发送，当这些分组到达目的结点后，就要将这些具有相同报文序号的多个分组重新组装成一个完整的报文再递交给目的端系统。若目的结点用于重装报文的缓冲区空间有限，而且它无法知道正在接收的报文究竟被拆成多少个数据分组，此时就可能发生死锁现象，因为它没办法完成整个分组的重装，在缓冲的分组就不能被发送出去。而此时相邻结点仍在不断地向它传送数据分组，因没有可用的缓冲空间，本结点就无法继续接收这些后续的分组了。虽然相邻结点可以重发，但目的结点的缓冲区始终得不到清空，这样源结点在经过多次尝试后，相邻结点就会绕道从其他途径再向该目的端系统传送数据分组，但这样做又会造成周边区域也由此发生拥塞。

以下几种方法可以避免重装死锁的发生：

- 允许目的结点将不完整的报文递交给目的端系统，只要按顺序发送即可，完整组装的任务交给目的端系统。
- 检测出不能完整重装的报文，然后要求发送该报文的源端系统重新传送，但这种方法的有效性比较低，因为这也很难确保目的结点在重发时就有足够的缓冲空间。
- 每个结点配备一个专门的缓冲空间，用以暂存不完整的报文，就像 CPU 上配置的多级缓存一样，但这明显增加了设备成本。

以上前两种方法不能很好地解决重装死锁，第一种方法可能使端系统接收到不完整的数据报而使整个数据无效；第二种方法使端系统中的协议复杂化了，同时加重了源端的负担；第三种方法虽然不涉及端系统，但却使每个结点增加了开销，系统性能会大受影响。

8.10 三层交换机与三层交换

早期的网络中一般使用二层交换机来搭建局域网，而不同局域网之间的网络互通由路由器来完成，如图 8-67 所示。那时的网络流量，局域网内部的流量占了绝大部分，而网络间的通信访问量比较少，使用少量路由器已经足够应付了。

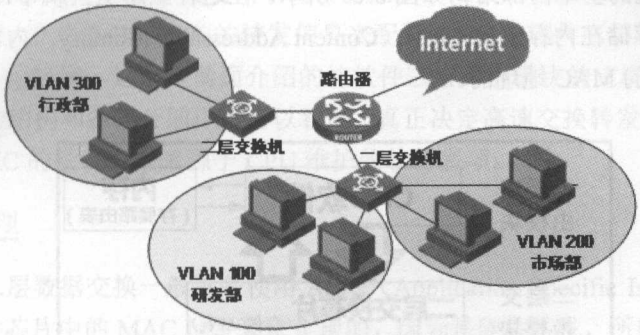


图 8-67 二层交换机+路由器组网基本结构

随着数据通信网络范围的不断扩大和网络业务的不断丰富，网络间互访的需求越来越大，而路由器由于自身成本高、转发性能低、端口数量少等特点无法很好地满足网络发展的需求。我们知道，路由器主要是通过 IP 转发（三层转发）来实现不同网络间的互连，那么是否能够将交换机的高性能应用到三层转发中去呢？答案是肯定的，三层交换机就是这样一种实现了高速三层转发的设备，其典型网络结构如图 8-68 所示。

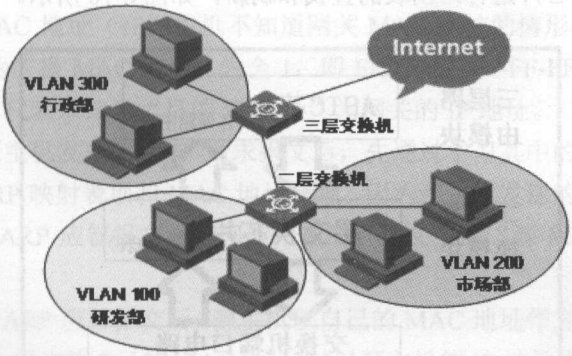


图 8-68 三层交换机组网基本结构

大多数三层交换机采用 ASIC 硬件芯片来完成转发，ASIC 芯片内部集成了 IP 三层转发的功能，包括检查 IP 报文头、修改存活时间（TTL）参数、重新计算 IP 头校验和、IP 数据报的数据链路封装等。

8.10.1 三层交换机硬件结构

总体来说，目前的三层交换机在三层路由模块中有 3 种类型：纯硬件模块、纯软件模块和软/硬件结合的模块。目前最常见的还是软/硬件结合的三层路由模块。

1. 纯软件路由模块的三层交换机

在早期以及目前比较低档的三层交换机中基本上都是采用纯软件路由模块来实现三层交换功能的。这类三层交换机的基本内部结构如图 8-69 所示。在这种三层交换机中，是通过 CPU 调用相关的软件功能，查询存储在内存中的 CAM（Content Addressable Memory，内容可寻址记忆）表来获取对应目的 IP 地址的 MAC 地址的。

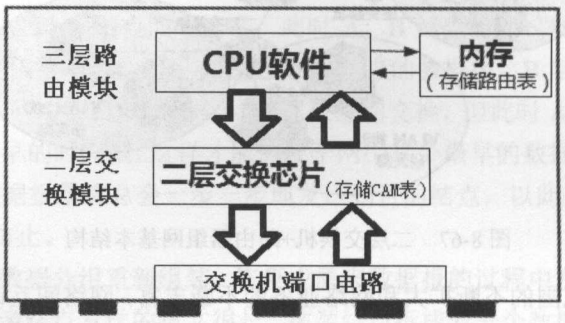


图 8-69 纯软件路由模块的三层交换机基本内部结构

2. 纯硬件路由模块的三层交换机

纯硬件的三层交换技术相对来说技术复杂、成本高，但是速度快、性能好、带负载能力强。其原理是采用专门的 ASIC 芯片进行路由表的查找和刷新，如图 8-70 所示。

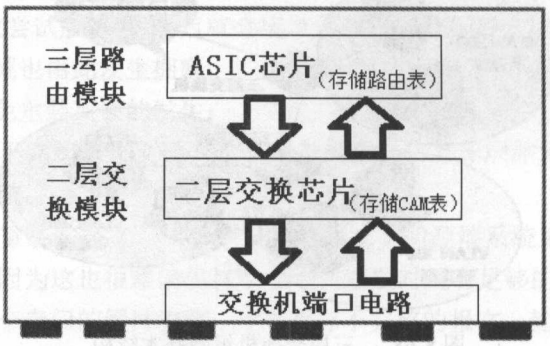


图 8-70 纯硬件路由模块的三层交换机基本内部结构

这种纯硬件路由模块的三交换机在数据交换原理方面与纯软件路由模块的三层交换机基本一样，不同的只是这里在进行路由转发时采用的是 ASIC 芯片，路由表信息也是存储在 ASIC 芯片中，

执行效率更高。

3. 软/硬件结合路由模块的三层交换机

目前纯软件或者纯硬件路由模块的三层交换机都比较少见，而更多的是采用两者的结合，在 ASIC 芯片中同时存储二层 CAM 表和三层转发表，在内存中存储软件路由表和 ARP 映射表，由 CPU 调用，既利用硬件模块的高速交换性能，又利用软件模块的灵活性。

以上 ASIC 芯片用来完成主要的二、三层转发功能，内部包含用于二层转发的 MAC 地址表以及用于 IP 转发的三层转发表；CPU 主要用于转发的控制，维护一些软件表项（包括软件路由表、软件 ARP 映射表等），并根据软件表项的转发信息来配置 ASIC 的硬件三层转发表。当然，CPU 本身也可以完成软件三层转发，这就是前面介绍的纯软件三层路由模块的三层交换机。

从三层交换机的结构和各部分的作用可以看出，真正决定高速交换转发的是 ASIC 中的二、三层硬件表项，而 ASIC 的硬件表项来源于 CPU 维护的软件表项。

8.10.2 三层交换原理

二层交换机的二层数据交换一般都是使用 ASIC（Application Specific Integrated Circuit，应用专用集成电路）硬件芯片中的 MAC 地址表来实现的，因为是硬件转发，所以转发性能非常高。而三层交换机的三层转发也是依靠 ASIC 芯片完成的（路由器的路由功能主要依靠 CPU 软件进行），但其中除了二层交换用的 MAC 地址表外，还保存有专门用于三层转发的三层硬件转发表。

三层交换的总体原理为：一次路由（第一个数据报由 CPU 的路由表转发并建立三层转发表项），多次交换（后续数据报由 ASIC 的三层转发表转发）。如图 8-71 所示是仅一个直接连接在一台三层交换机上的两个不同网段主机三层交换的基本流程，各主要步骤解释如下：

（1）源主机在发起通信之前，将自己的 IP 地址与目的主机的 IP 地址进行比较，如果源主机判断目的主机与自己位于不同网段时，它需要通过网关来递交报文，所以它首先需要通过一个 ARP 请求报文获取网关的 MAC 地址（在源主机不知道网关 MAC 地址的情形下）。ARP 请求帧的帧头部“目的 MAC 地址”为广播 MAC 地址，为全 1，即 FF-FF-FF-FF-FF-FF，“数据”字段中的“目的 MAC 地址”为全 0（代表未知），“目的 IP 地址”为网关的 IP 地址。

（2）网关在收到源主机发来的 ARP 请求报文后，先通过学习其中的源 MAC 地址和源 IP 地址建立基于源主机的 ARP 映射表项和 MAC 地址表项。因为源主机发送的 ARP 请求报文是给网关的，所以网关会以一个 ARP 应答报文进行回应，在应答报文中的“源 MAC 地址”就包含了网关的 MAC 地址。

（3）在得到网关的 ARP 应答报文后，源主机以自己的 MAC 地址作为帧中的“源 MAC 地址”，以网关的 MAC 地址作为报文的“目的 MAC 地址”，以源主机的 IP 地址作为报文的“源 IP 地址”，以目的主机的 IP 地址作为“目的 IP 地址”，先把发送给目的主机的数据发给网关。

（4）网关在收到源主机发送给目的主机的数据后，由于查看得知源主机和目的主机的 IP 地址不在同一网段，而且虽然帧头部分“目的 MAC 地址”是网关自己的，但“目的 IP 地址”不是自己的（这是判断是否要对数据包进行三层转发的依据），得知这个数据不是发给自己的，知道此数

据要由自己进行转发。于是，先把数据上传到三层交换引擎（ASIC 芯片），在里面查看有无基于目的主机 IP 地址的三层硬件转发表。

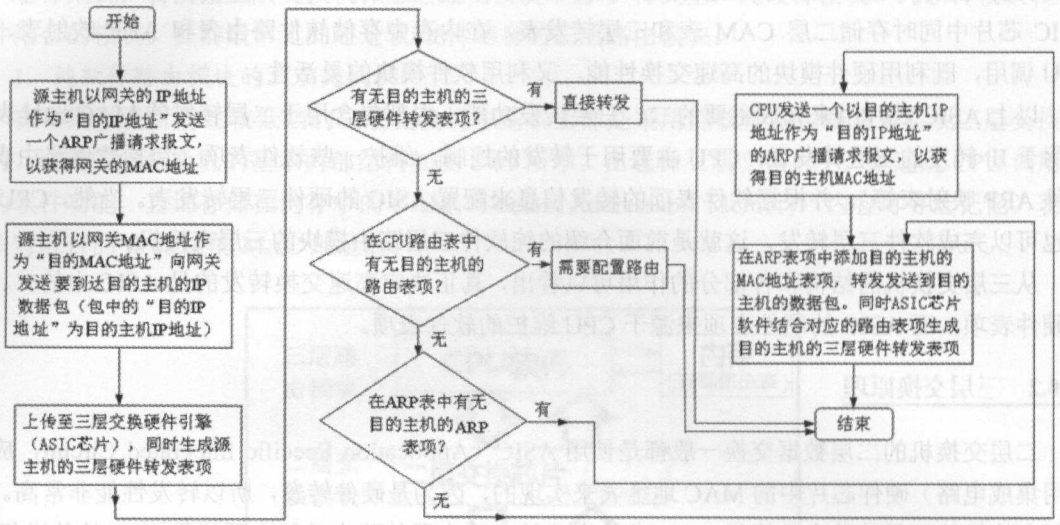


图 8-71　三层交换基本流程

（5）如果在三层硬件转发表中没有找到目的主机的对应表项，则向 CPU 请求查看软件路由表，如果有目的主机所在网段的路由表项，则还需要得到目的主机的 MAC 地址，因为数据报在链路层是要经过帧封装的。于是三层交换机 CPU 根据路由表项所对应的出接口向目的主机所在网段发送一个 ARP 广播请求包，以获得目的主机 MAC 地址。

（6）目的主机在收到网关发送的 ARP 请求报文后，因为发现报文中的“目的 IP 地址”是自己的，于是接收，并向网关发送一个 ARP 响应报文。

（7）三层交换机在获得目的主机 MAC 地址后，同样先会通过学习报文中的“源 MAC 地址”和“源 IP 地址”建立基于目的主机的 ARP 映射表项和 MAC 地址表项，然后用所得到的目的主机 MAC 地址对原来接收的来自源主机的数据进行重新帧封装，帧中“源 MAC 地址”为发送该帧出接口的 MAC 地址，“源 IP 地址”为出接口 IP 地址，“目的 IP 地址”为目的主机的 IP 地址，“目的 MAC 地址”为目的主机的 MAC 地址，然后转发给目的主机。同时三层交换机的三层引擎会结合路由表生成基于目的主机的三层硬件转发表。

以后到达目的主机的数据报就可以直接利用三层硬件转发表中的转发表项进行数据交换，不用再查看 CPU 中的路由表了。这就是前面所说的“一次路由，多次交换”的意思。

8.10.3　三层交换示例

在三层交换中，同一交换机上的不同网段主机通信和不同交换机上的不同网段主机通信的基本原理是一样的，只是具体流程有所区别。本节仅以比较简单的“同一交换机上的不同网段主机通信”

这种情形来解释上节介绍的三层交换原理。

如图 8-72 所示，通信的源、目的主机连接在同一台三层交换机上，但它们位于不同的 VLAN（也位于不同网段）。对于三层交换机来说，这两台主机都位于它的直连网段内，它们的 IP 对应的路由都是直连路由。图中已标明了两台主机的 MAC 地址、IP 地址、网关 IP 地址（也就是对应 VLAN 接口 IP 地址），以及三层交换机的 MAC 地址。

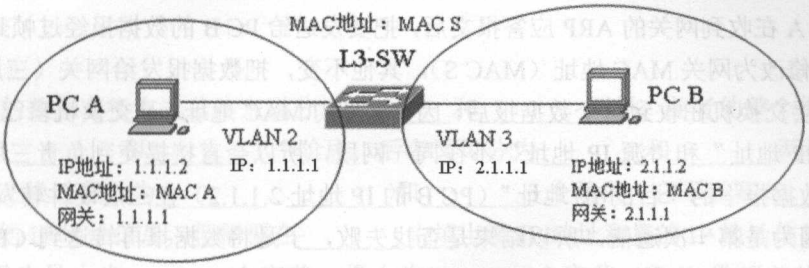


图 8-72 同一交换机上不同网段主机间的三层交换示例

【说明】本示例中虽然划分了 VLAN，但是在下面的数据转发流程中我们并没有提到 VLAN 标记，那是因为在本示例中，通信双方主机都连接到同一个三层交换机上，端口类型均为 Access 类型，发送和接收的数据帧都是无 VLAN 标记的。

当 PC A 要第一次向 PC B 发向数据报时，数据报的传输流程如下（假设三层交换机上当前还未建立任何硬件转发表项）：

（1）PC A 先检查出目的 IP 地址 2.1.1.2（PC B）与自己不在同一网段，觉得就这样直接发送肯定不行，于是把要发送的数据报先缓存起来。它知道这个数据报必须经过网关来转发，所以先查看一下自己是否已有网关的 MAC 地址（也就是在 PC A 主机的 ARP 映射表中查看是否有与网关 IP 地址对应的 MAC 地址表项）。如果有，则直接把要发送给 PC B 的数据报在封装成数据帧时把“目的 MAC 地址”字段的值设为网关的 MAC 地址（也就是三层交换机的 MAC 地址 MAC S，交换机上各端口、各 VLAN 是共享一个或多个 MAC 地址的）。

（2）如果 PC A 在自己的 ARP 映射表中没有找到网关 MAC 地址，则先向网关发出（其实是会向本 VLAN 内的所有结点发出）一个 ARP 广播请求报文，以获取网关 IP 地址 1.1.1.1 所对应的 MAC 地址。此 ARP 请求报文的“源 MAC 地址”字段是 PC A 的 MAC 地址 MAC A，“目的 MAC 地址”因为未知，以全 0 格式填充；“源 IP 地址”和“目的 IP 地址”字段分别填上 PC A 的 IP 地址（1.1.1.2）和网关的 IP 地址（1.1.1.1）。

ARP 请求报文向下传输到了以太网数据链路层后被再次封装成以太网帧，以太网帧头中的“源 MAC 地址”字段值仍为 PC A 的 MAC 地址 MAC A，“目的 MAC 地址”字段值为广播 MAC 地址 FF-FF-FF-FF-FF-FF，“帧类型”字段填上 ARP 的协议号 0x0806。有关 ARP 报文和 ARP 帧格式参见 8.4.1 节。

（3）三层交换机在收到 PC A 发来的 ARP 请求报文后，检查请求报文发现被请求 IP 地址（也

就是“目的 IP 地址”）是自己的三层接口 IP 地址，于是向 PC A 发回一个 ARP 应答报文，并将对应的网关接口 MAC 地址（MAC S）填充在应答报文中的“源 MAC 地址”字段中。同时通过对 PC A 发送的 ARP 请求报文分析把 PC A 的 IP 地址与 MAC 地址的对应关系（1.1.1.2<==>MAC A）记录到自己的 ARP 映射表中去，然后把 PC A 的 IP 地址（作为“目的 IP 地址”）、MAC 地址（作为“下一跳 MAC 地址”），以及与交换机直接相连的端口号等信息下发到三层交换机 ASIC 芯片中的三层硬件转发表中。此时在三层硬件转发表中就有了第一个转发表项，即 PC A 的转发表项。

（4）PC A 在收到网关的 ARP 应答报文后，把要发送给 PC B 的数据报经过帧封装后的“目的 MAC 地址”修改为网关 MAC 地址（MAC S），其他不变，把数据报发给网关（三层交换机）。

（5）三层交换机在收到这个数据报后，因为“目的 MAC 地址”是交换机自己的 MAC 地址，而且“目的 IP 地址”和“源 IP 地址”不在同一网段，所以会直接提交到负责三层交换的 ASIC 芯片，根据数据报中的“目的 IP 地址”（PC B 的 IP 地址 2.1.1.2）在三层硬件转发表中查看有无对应表项，因为是第一次通信，所以结果是查找失败，于是将数据报再转送到 CPU 去进行软件路由处理。

（6）CPU 同样会根据数据报中的“目的 IP 地址”去查找其软件路由表，发现匹配了一个直连网段（PC B 对应的网段），于是继续在 ARP 映射表中查找对应的 MAC 地址项。同样由于是第一次查找，所以仍然查找失败。如果在 ARP 映射表中找到了对应的 MAC 地址，则数据可以直接由软件路由表转发了。

（7）下面仍以在 ARP 映射表中也没找到 PC B 对应的 MAC 地址为例进行介绍。此时三层交换机 CPU 会在 PC B 所在网段的 VLAN 3 中的所有端口发送一个 ARP 广播请求报文，查找“目的 IP 地址”为 2.1.1.2 所对应的 MAC 地址。报文经过帧封装后的“源 MAC 地址”是三层交换机的网关 MAC 地址（MAC S），“目的 MAC 地址”全为 0，“源 IP 地址”是 VLAN 3 网段的网关 IP 地址（2.1.1.1），“目的 IP 地址”是 PC B 的 IP 地址（2.1.1.2）。

（8）PC B 在收到三层交换机 CPU 发送的 ARP 请求报文后，检查发现被请求的 IP 地址是自己的 IP 地址，于是发送一个 ARP 应答报文，并将自己的 MAC 地址（MAC B）包含在其中。同时，将三层交换机上 VLAN 3 网段的网关 IP 地址与 MAC 地址的对应关系（2.1.1.1<==>MAC S）记录到自己的 ARP 映射表中去。

（9）三层交换机 CPU 在收到 PC B 的 ARP 应答报文后，将其 IP 地址和 MAC 地址对应关系（2.1.1.2<==>MAC B）记录到自己的 ARP 映射表中去，并把 PC B 的 IP 地址、MAC 地址、进入交换机的端口号等信息下发到三层交换机的三层转发表中。此时转发表中就有到达 PC A 和 PC B 这两条对应的表项了。

（10）三层交换机 CPU 根据获得的 PC B 的 MAC 地址和端口信息，以及软件路由表信息，把由 PC A 发来的 IP 数据报转发给 PC B，这样就完成了 PC A 到 PC B 的第一次单向通信。

由于芯片内部的三层引擎中已经保存了从 PC A 到 PC B 的完整转发路径信息，所以以后 PC A 与 PC B 之间进行通信，或其他网段的站点想要与 PC A 或 PC B 进行通信时，三层交换机的 ASIC 芯片就会直接把数据报从对应的三层硬件转发表项中指定的端口转发出去，而不必再把包交给

CPU 进行路由处理。这就是所谓的“一次路由（指通过 CPU 路由表查到了对应的直连网段），多次交换”的原理，大大提高了转发速度。

8.11 课后自测题

为了巩固本章所学的内容，请认真做下面的自测题，参考答案见附录。

8.11.1 填空题

1．网络层的数据交换方式总体来说分为_____和_____两类，前者的整个数据交换过程分为_____、_____和_____3 个阶段，后者又可分为_____和_____两种。

2．虚电路交换过程分为_____、_____和_____3 个阶段，它不需要在_____中填上源 IP 地址和目的 IP 地址信息，但需要在_____中填上源 IP 地址和目的 IP 地址信息。

3．IPv4 报头最小为_____个字节，最大为_____个字节。IPv4 数据包的最大长度为_____个字节，但其必须是_____的整数倍，否则要在_____以 0 填充。

4．IPv6 报头除了有_____外，还可以有多个_____，前者的总长度是固定为_____个字节。总的来说，IPv6 基本报头去掉了 IPv4 报头中的_____、_____、_____、_____、_____、_____和_____字段，却只增加了一个_____字段。

5．IPv6 扩展报头类型主要有_____、_____、_____、_____、_____、_____和_____。

6．网络层发送的 ARP 请求报文中的“目的 MAC 地址”字段值固定为（均用十六进制表示）_____，“协议类型”字段在映射 IP 协议时固定为_____；而在数据链路层封装成 ARP 请求帧后，帧头中的“目的 MAC 地址”固定为_____，“类型”字段固定为_____，表示它的上层协议就是 ARP 协议。

7．ARP 表项是网络设备_____和_____的映射（绑定）关系，其目的是通过设备的_____解析其对应的_____。ARP 表项可分为_____和_____两类，前者又可分为_____和_____两种，其中前者除了有普通 ARP 映射表项的绑定关系外，还同时绑定了_____和_____。

8．路由算法总体上可分为“非自适应路由算法”和“自适应路由算法”两大类，前者主要包括_____、_____、_____、_____和_____；后者主要包括_____、_____和_____3 种。

9．每个静态路由表项至少应包括_____、_____和_____3 个信息。

10．在扩散路由算法中，为了解决各端口会重复接收相同分组的问题采用了_____和_____两种方法。

11．网络拥塞控制方法主要有_____、_____和_____3 种。“死锁”是一些结点由于_____，死锁的类型有_____和_____两种，前者有两种预防的方法，分别是_____和

_____。

12. 在三层交换中主要用到 3 张表，先根据数据包中的目的 IP 地址查看有无对应的_____，如果没有再继续查看有无对应的_____，如果还没有则查看有无对应的_____。

8.11.2 选择题（可多选）

1. 下列数据单元不包括源地址和目的地址信息的是（　　）。

A. 报文　　B. 报文分组　　C. 数据报分组　　D. 虚电路分组

2. 下列数据交换方式中不需要依靠路由技术的是（　　）。

A. 线路交换　　B. 报文交换

C. 数据报分组交换　　D. 虚电路分组交换

3. 下面关于虚电路交换和数据报交换说法不正确的是（　　）。

A. 虚电路交换中分组的头部是不带有 IP 地址信息的，数据报交换中是有的

B. 虚电路交换需要事先建立通信连接，数据报交换不需要

C. 虚电路交换中同一次通信发送的所有分组都沿着固定的路径转发

D. 虚电路交换比数据报交换具有更高的拥塞控制能力，通信效率更高

4. 在图 8-73 中，假设 R1 中已创建了从 PC1 到 PC3、从 PC2 到 PC3、从 PC3 到 PC4 的 3 条 VC，并且所分配的 VC 号分别为 VC1、VC2、VC3；R3 中已创建一条从 PC3 到 PC4 和从 PC2 到 PC4 的 VC，并且所分配的 VC 号分别为 VC3、VC4。现如果 PC1 再要通过 R1、R3 创建一条到达 PC4 的 VC，则这个 VC 号可以是（　　）。

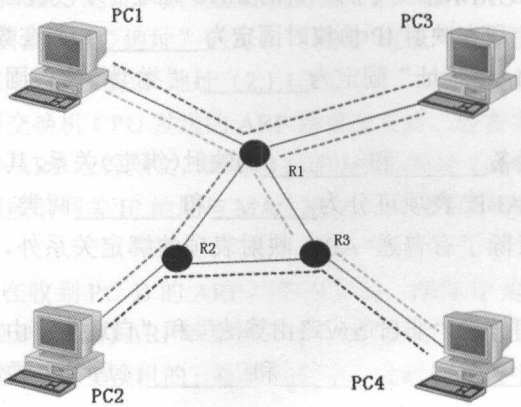


图 8-73　第 4 题图

A. VC1　　B. VC2　　C. VC3

D. VC4　　E. VC5

5. 下列 IPv6 扩展头在一个 IPv6 数据包中可以出现两次的是（　　）。

A. 逐跳选项头　　B. 目的选项头　　C. 路由头

D．分段头　　E．封装安全有效载荷头

6．下列 IPv6 扩展报头必须在所有 IPv6 扩展报头前面的是（　）。

A．逐跳选项头　　B．目的选项头　　C．路由头

D．分段头　　E．认证头

7．以太网链路中来自网络层的 IPv4 数据报长度不能大于（　）个字节，否则要被拆分。

A．1492　　B．1500　　C．1518　　D．65535

8．下列关于静态 ARP 映射表项和动态 ARP 映射表项的说法不正确的是（　）。

A．静态 ARP 映射表项只能手工创建，动态 ARP 映射表项都不通过手工创建

B．静态 ARP 映射表项不会被老化，但可以被动态 ARP 映射表项覆盖

C．动态 ARP 映射表项会被老化，也可以被静态 ARP 映射表项覆盖

D．在相应设备接口关闭时，与该接口相关的 ARP 映射表项都被删除

9．下列协议报文中不封装 IPv4 头部的是（　）。

A．ARP　　B．ICMP　　C．IGMP　　D．ND

10．当发送的是 ICMP 报文时，在 IPv4 头部中的“协议”字段值为（　）。

A．0x01　　B．0x06　　C．0x17　　D．0x18

11．如果返回到 ICMP 报文中的“类型”值为 3，则含义不可能是（　）。

A．不能访问主机　　B．无法访问端口　　C．需要重定向　　D．无法访问协议

12．Ping 某台主机时，如果要证明该主机存在，则返回的 ICMP 报文中“类型”字段的值为（　）。

A．8　　B．4　　C．3　　D．0

13．在下列 IPv6 协议簇中用来取代 IPv4 协议簇中的 IGMP 功能的是（　）。

A．ICMPv6　　B．IGMPv6

C．MLD　　D．ND

14．下列功能中不属于 ND 功能的是（　）。

A．邻居路由器的发现　　B．数据报无法传送报告

C．解析一个相邻结点的链路层地址　　D．发现并自动配置地址

E．动态公布 MAC 地址的更改　　F．确定某个相邻结点是否可达

15．在 Cisco 和华为/H3C 设备中，静态路由的默认优先级分别为（　）。

A．0　1　　B．0　10

C．1　10　　D．1　60

16．在 Cisco 和华为/H3C 设备中，RIP 路由的默认优先级分别为（　）。

A．100　10　　B．10　100

C．120　100　　D．100　120

17．下列路由协议中使用距离矢量路由算法的是（　）。

A．RIP　　B．IGRP　　C．EIGRP

D. OSPF　　E. IS-IS

18. 下列路由协议中使用链路状态路由算法的是（　）。

A. RIP　　B. IGRP　　C. EIGRP

D. OSPF　　E. IS-IS

19. 下列路由协议中使用了 Dijkstra 算法的是（　）。

A. RIP　　B. IGRP　　C. EIGRP

D. OSPF　　E. IS-IS

20. 下面（　）是发生重装死锁的原因。

A. 目的结点没有足够的用于数据重装的缓存空间

B. 接收到的某个数据报所拆分的分组太多

C. 发送方发送数据的速率太快

D. 发送方一次性连续发送的数据太多，没有足够的缓存空间

21. 数据包分片可以在任何设备上发生，但分片组装只能在（　）进行。

A. 报文环回时　　B. 下一台路由器

C. 所有中间路由器　　D. 报文目的地

22. 三层交换机在对数据进行三层交换时是有一个判断依据的，它是（　）。

A. 本地 ARP 表中是否有基于数据包中目的 IP 地址的 ARP 表项

B. 本地路由表中是否有基于数据包中目的 IP 地址的路由表项

C. 本地三层转发表中是否有基于数据包中目的 IP 地址的转发表项

D. 帧头部“目的 MAC 地址”是网关的，但数据包头部“目的 IP 地址”不是网关的

23. 关于免费 ARP 报文说法不正确的是（　）。

A. 通过发送免费 ARP 报文可以确认 IP 地址是否有冲突

B. 免费 ARP 报文格式属于 ARP 应答报文

C. 免费 ARP 报文可帮助更新旧的 IP 地址信息

D. 免费 ARP 报文有助于防止网关 ARP 攻击

24. 下面关于代理 ARP 功能说法正确的是（　）。

A. 代理 ARP 可以连接两个在不同物理网段，但却在同一个 IP 网段或同一个汇聚网段的网络

B. 要启用代理 ARP 功能，主机上必须先配置好网关 IP 地址

C. 启用代理 ARP 功能后，源主机在没有获得目的主机 MAC 地址时先要向网关发送一个 ARP 请求报文，以获得网关的 MAC 地址

D. 启用代理 ARP 功能后，网关会替代源主机在其他物理网段查找目的主机的 MAC 地址，并且告诉源主机

25. 关于图 8-74 所示拓扑结构（交换机为二层交换机）描述正确的是（　）。

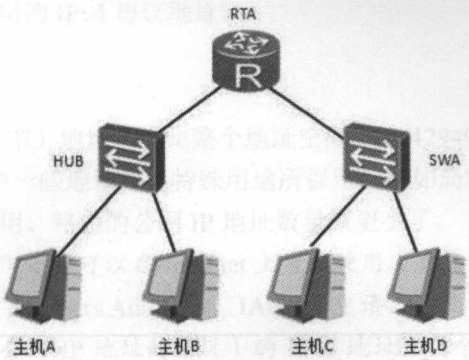


图 8-74 拓扑结构图

A．此网络中有 6 个冲突域　　B．此网络中有 2 个广播域

C．此网络中有 4 个冲突域　　D．此网络中有 6 个广播域

26．关于 IPv4 报头中“总长度”字段说法正确的是（　　）。

A．表示整个 IPv4 报文头部的总长度　　B．以字节为单位

C．必须是 4 个字节的整数倍　　D．最小为 20 个字节，最大为 60 个字节

27．已经分段过的数据报有可能会再次进行分段。IP 头部中包含的（　　）为分段和重新组装提供了足够的信息。

A．区分服务　　B．标志　　C．段偏移　　D．头部长度

28．下面关于 IPv4 报头“标志”字段和“段偏移”字段说法错误的是（　　）。

A．“标志”字段的中间一位为 0 时表示后面没有分段，为 1 时表示有分段

B．“标志”字段最低一位为 1 时表示后面还有分段

C．“段偏移”字段是以 8 个字节为单位进行计算的

D．“段偏移”字段指出的是本分组中“数据”部分的起始位置相对原 IPv4 报文“数据”字段起始位置的偏移

29．下面关于 IPv6 报头说法正确的是（　　）。

A．“有效载荷长度”字段是以字节为单位进行计算的，不包括扩展报头部分

B．IPv6 报头长度为 20 个字节

C．每个扩展头长度不限，但都是 8 个字节的整数倍

D．每个扩展报头均可在一个 IPv6 报文中出现多次

9

IP 地址和子网

通过上一章我们已对 IPv4 和 IPv6 协议有了一个比较全面的了解，重点学习了它们各自的报文格式、IPv4 报文封装和解封装、IPv4 报文分段与重组，以及 IPv4 和 IPv6 协议簇中主要的子协议用途、报文格式等。

本章将专门介绍在 IP 网络中我们每天都要配置的 IPv4 和 IPv6 地址，其中主要包括 IPv4 地址和 IPv6 地址的表示形式与分类，各类 IPv4、IPv6 地址结构，以及在进行网络规划和设计中经常需要考虑的 IPv4 子网划分与聚合方法等。这些都是非常基础但却非常重要的知识和技能，不要说网络职业人士，就是其他专业的 IT 职业人员都必须有比较全面的了解和掌握，因为无论是在一般的计算机配置中，还是在网络规划、设计与组建过程中，IP 地址的配置与管理都是必不可少的。

本章重点与难点：

- IPv4 地址分类及各类地址的主要特点
- IPv4 网络地址、广播地址和主机地址
- IPv4 子网划分与聚合的方法
- IPv6 地址表示形式
- IPv6 地址分类及各类地址的主要特点

9.1 IPv4 地址

在计算机网络体系结构“网络层”最重要的一个协议就是 IP 协议，目前正处于 IPv4 和 IPv6 这两个版本交接、过渡时期，所以本章会同时对这两个版本的 IP 地址及相关知识进行系统的介绍，

本节先来介绍目前仍主流使用的 IPv4 协议地址。

9.1.1 IPv4 地址基本格式

IPv4 使用 32 位（4 个字节）地址，因此整个地址空间中有 4294967296（2^{32}）个地址，也就是近 43 亿个地址。不过，其中一些地址是为特殊用途所保留的，如局域网专用地址和组播地址，这样一来可直接在广域网中使用、路由的公网 IP 地址数量就更少了。

【说明】“公网 IP 地址”是指可以在 Internet 上直接使用、直接被路由，并需要向互联网数字分配机构（Internet Assigned Numbers Authority，IANA）申请、注册、购买，且全球唯一（不存在多个用户拥有、使用相同的公网 IP 地址的情况）的 IPv4 地址。打个比方来说，公网 IP 地址就相当于我们公民的身份证号，每个身份证号都是全国唯一的，并且通过这个身份证号就可以查到我们的基本信息。公网 IP 地址可以直接分配给互联网上的主机、服务器或其他设备，这样可以通过它在全球范围内找到对应的主机、服务器和设备，因为这些公网 IP 地址都是需要在互联网管理机构注册的。如各大企业网站通常都是直接使用公网 IP 地址的。

与“公网 IP 地址”相对应的自然是“私网 IP 地址”，也称“专用网络 IP 地址”或者“局域网 IP 地址”。私网 IP 地址是指仅可以在各用户自己的局域网内部使用，且不同用户可以重复使用，无需向 IANA 申请、注册、购买的 IPv4 地址。私网 IP 地址就相当于我们企业内部的员工编号，仅在企业内部有用，不能通过这个员工编号来在全国范围内找到我们。企业内部局域网使用的就是私网 IP 地址，具体有哪些 IP 地址属于私网 IP 地址将在本章后面介绍。

随着公网 IP 地址不断被分配给最终用户，IPv4 地址枯竭问题也在随之产生。虽然基于可变长子网掩码（VLSM）、无类别域间路由（CIDR）和网络地址转换（NAT）的地址结构重构显著地降低了地址枯竭的速度，但在 2011 年 2 月 3 日，在最后 5 个地址块被分配给 5 个区域互联网注册管理机构之后，IANA 的主要地址池空了，所以才有了现在正积极推行的 IPv6 版本，具体将在本章后面介绍。

IPv4 地址在计算机内部是以二进制形式表示，每个地址都是 32 位（4 个字节），由数字 0 或 1 构成，每 8 位（1 个字节）为一段（假设分别用 w、x、y、z 表示，但在计算机内部这 4 段之间并没有我们所看到的那个用来分隔各段的一个小圆点，只是为了方便我们自己分辨而在每个字节间用一个小圆点分隔），如图 9-1 所示。

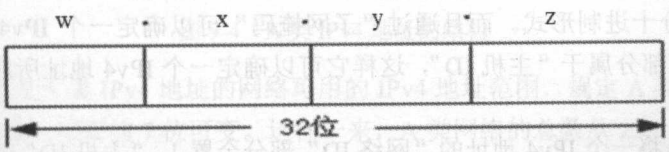


图 9-1 点分二进制表示形式的 IPv4 地址

由于整个 IPv4 地址有 32 位长，无论是书写还是记忆都不方便，于是我们在日常的 IPv4 地址使用中就把这 4 段转换成对应的十进制，同样在每个字节间用小圆点分隔，每段最多只需用 3 位十

进制数（8 位所能表示的最大十进制数为 $255=2^8-1$）表示，这样就简单、明了了许多。

32 位二进制 IPv4 地址又分成了两部分：一部分用来表示此 IPv4 地址所属的网段（一个标准网络或子网，下同），称为“网络 ID”（Network ID）；另一部分是用来表示具体主机或节点的 IPv4 地址，称为“主机 ID”（Host ID），如图 9-2 所示。具体网络 ID 和主机 ID 各占多少位要视所属的网络而定，具体将在本章后面介绍。

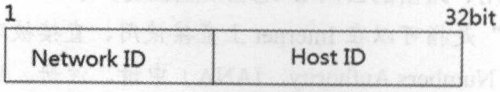


图 9-2　IPv4 地址的两个部分

9.1.2　几个重要概念

在配置 IPv4 地址时涉及到几个重要概念，即网络地址、广播地址和子网掩码。

1. 网络地址

“网络地址”是用来标识一个网段的，是将 IPv4 地址中的“主机 ID”部分全置 0，而“网络 ID”部分保持不变得到的。它是一个网段中的第一个 IP 地址，不可分配给主机或节点使用。在路由表项中“目的网络”部分就是用对应网段的“网络地址”来标识的。

2. 广播地址

“广播地址”是用来进行广播传输的 IPv4 地址，同时也标识一个广播域，是将 IPv4 地址中的“主机 ID”部分全置 1，“网络 ID”部分保持不变得到的。它是一个网段中的最后一个 IPv4 地址，也不可分配给主机或节点使用。“目的 IP 地址”为广播地址的数据报仅可在该广播地址所代表的广播域内传播。

3. 子网掩码

我们在为设备配置 IP 地址时，通常不能仅配置 IPv4 地址，而必须同时配置“子网掩码”，如图 9-3 所示。那么“子网掩码”是什么？它又有什么作用？

子网掩码的作用就是获取主机 IPv4 地址中的网络地址信息，用于区别主机通信的不同情况，选择不同的路由。子网掩码一旦设置，对应 IPv4 地址中的“网络 ID”和“主机 ID”部分就固定了。虽然子网掩码不是一个 IPv4 地址，但它与 IPv4 地址一样也是 32 位二进制、分成 4 段，当然也可以转换成点分十进制形式。而且通过“子网掩码”可以确定一个 IPv4 地址的哪一部分属于“网络 ID”，哪一部分属于“主机 ID”，这样它可以确定一个 IPv4 地址所属的网络和对应的网络地址。

“子网掩码”是将一个 IPv4 地址的“网络 ID”部分全置 1，“主机 ID”部分全置 0 得到的。通过把一个 IPv4 地址与其对应的“子网掩码”进行逻辑“与”运算，就得到了该网络的“网络地址”。具体子网掩码的计算要视具体的网络或子网而定，将在本章后面介绍。

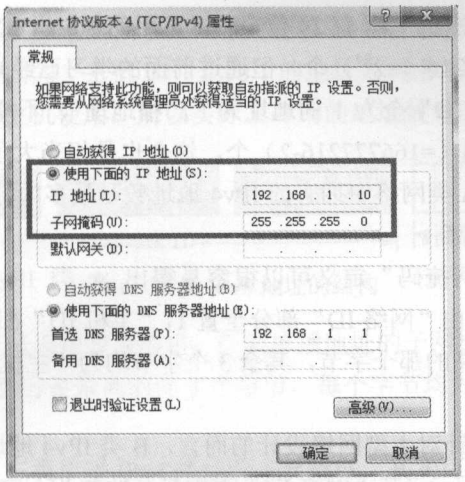


图 9-3 同时配置的 IPv4 地址和子网掩码

9.1.3 IPv4 地址分类

最初的 IPv4 网络比较少，那么多 IPv4 地址完全够用，所以那时根本没有“子网”的概念，首先根据网络通信类型分成了单播地址和组播地址两大类，然后在单播地址类别中又根据网络规模大小分成了三大类，另外为了预留一些地址又单独划分了一类。这样就一共有 5 类，分别用 A、B、C、D 和 E 表示。其中 A、B、C 这 3 类是单播通信地址类型，D 是组播通信地址类型，E 是保留地址类型。当然，这五大类的地址范围和定义肯定不是随意的，下面进行详细介绍。

1. A 类 IPv4 地址

A 类 IPv4 地址是专门为网络规模比较大的网络而设计的单播 IPv4 地址分类，因为它的“网络 ID”部分所占的位数最多，自然包括的 IPv4 地址数是最多的。A 类 IPv4 地址结构如图 9-4 所示，用来标识网络的“网络 ID”部分只占 IPv4 地址中的最高 1 个字节，“主机 ID”占用了剩余的全部 3 个字节。

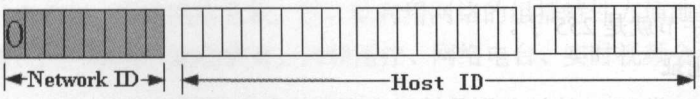


图 9-4 A 类 IPv4 地址的结构

另外为了限制使用 A 类 IPv4 地址的网络可用的 IPv4 地址范围，规定 A 类 IPv4 地址中“网络 ID”的最高位固定为 0，后面的 7 位可变。这样一来，A 类网络的总数从 256（$=2^8$）个减少到 128（$=2^7$）个。但实际上可用的只有 126 个，即整个 IPv4 地址中可以构建 126 个 A 类网络，因为“网络 ID”为 0 和 127 的 A 类网络不可用。“网络 ID”全为 0 的地址被保留，不能分配；而“网络 ID”为 01111111（相当于十进制的 127）的地址专门用于本地环路测试（也就是通常所说的环路地址），也是不能被分配的。也就是凡是以 0 或者 127 开头的地址都是不能分配给节点使用的。

因为 A 类 IPv4 地址中“主机 ID”有 24 位，所以可用的“主机 ID”数也就是每个 A 类网络中拥有的 IPv4 地址数为 166777216（$=2^{24}$）个。但通过前面的学习已经知道，“主机 ID”全为 0 的地址为“网络地址”，而“主机 ID”全为 1 的地址为“广播地址”，不能分配给主机使用，所以实际上可用的地址数为 166777214（=166777216-2）个，这个也是相当大了。所以 A 类 IPv4 地址包含的网络数是最少的，但每个 A 类网络中拥有的 IPv4 地址数是最多的，也就是可以构建的网络规模最大，适用于大型企业和运营商。

再根据上节介绍的“子网掩码”定义可以很容易得出 A 类 IPv4 地址的子网掩码为固定的 255.0.0.0，因为子网掩码就是由“网络 ID”部分全置 1，“主机 ID”部分全置 0 得到的，而 A 类地址中“网络 ID”部分就是最高的那个字节，其余 3 个字节均为“主机 ID”部分。

2. B 类 IPv4 地址

相比于 A 类 IPv4 地址是针对大型网络设计的而言，B 类 IPv4 地址是针对中型网络而设计的。B 类 IPv4 地址结构如图 9-5 所示，其“网络 ID”占用最高的前两个字节，而“主机 ID”则占用剩余的低两个字节。

同样为了限制使用 B 类 IPv4 地址的网络可用的 IPv4 地址范围，规定 B 类 IPv4 地址“网络 ID”的最高两位固定分别为 1、0，后面的 14 位可变。由此可知 B 类网络的总数从 65536（$=2^{16}$，也可写成 256×256）个减少到 16384（64×256=16384）个；B 类 IPv4 地址中“主机 ID”为 16 位，所以可用的“主机 ID”数也就是每个 B 类网络拥有的 IPv4 地址数为 65536（$=2^{16}$）个。同样因为“主机 ID”全为 1 的地址为网络地址，而“主机 ID”全为 1 的地址为广播地址，均不能分配给主机使用，所以实际上可用的地址数为 65534（=65536-2）个。

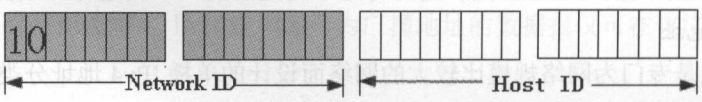


图 9-5　B 类 IPv4 地址的结构

再根据上节介绍的“子网掩码”定义可以很容易得出 B 类 IPv4 地址的子网掩码为固定的 255.255.0.0，因为 B 类地址中“网络 ID”部分是最高的两个字节，每个字节均为 8 个连续的 1，转换成十进制后每个字节就是 255 了。

3. C 类 IPv4 地址

前面的 A 类和 B 类 IPv4 地址分别是针对大型网络和中型网络而设计的，自然 C 类 IPv4 地址是针对小型网络而设计的。C 类 IPv4 地址结构如图 9-6 所示，其“网络 ID”占用最高的前 3 个字节，而“主机 ID”只占用最后的一个字节。从中可以得出，采用 C 类 IPv4 地址的网络数是最多的，但每个 C 类 IPv4 网络中可使用的 IPv4 地址数又是最少的，这正好符合中小型企业占大多数，而每个中小型企业网络中的用户数又不多的特点。

C 类 IPv4 地址“网络 ID”的最高 3 位固定分别为 1、1、0，后面的 21 位可变。由此得知 C 类网络总数从 166777216（$=2^{24}$，也可以写成 256×256×256）个减少到 2097152（32×256×256）个。

C 类 IPv4 地址中“主机 ID”仅为 8 位，所以可用的“主机 ID”数也就是每个 C 类网络拥有的 IPv4 地址数为 256（=2^8）个。同样因为“主机 ID”全为 1 的地址为网络地址，而“主机 ID”全为 1 的地址为广播地址，不能分配给主机使用，所以实际上可用的地址数为 254（=256-2）个。

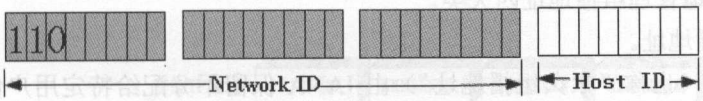


图 9-6 C 类 IPv4 地址的结构

同样根据子网掩码的定义可以很容易得出，C 类单播地址的子网掩码为固定的 255.255.255.0，因为 C 类地址中“网络 ID”部分是最高的前 3 个字节，每个字节均为 8 个连续的 1，转换成十进制后每个字节就是 255 了。

表 9-1 总结了 A、B 和 C 三类单播 IPv4 地址的主要特征，其实可以直接从第一个字节 w 的取值得出某个 IPv4 地址是属于哪类网络中的 IPv4 地址了。

表 9-1 A、B 和 C 三类 IPv4 单播地址的主要特征

类别	w 的值	网络 ID	主机 ID	可用网络数	每个网络可用的主机地址数
A	1～126	w	x.y.z	126	16777214
B	128～191	w.x	y.z	16384	65534
C	192～223	w.x.y	z	2097152	254

4. D 类 IPv4 地址

前面介绍的 A、B、C 类 IPv4 地址均是应用于单播通信的单播 IPv4 地址，此处介绍的 D 类 IPv4 地址是应用于组播通信的，是组播 IPv4 地址。通过组播 IPv4 地址，组播源（配置的 IP 地址仍是单播 IPv4 地址）只需发送一份数据就可以使对应组播组（以 D 类组播地址标识）的所有用户均可收到同一份数据。

IP 组播技术有效地解决了单点发送多点接收的问题，实现了 IP 网络中点到多点的高效数据传送，能够大量节约网络带宽、降低网络负载。还可以利用网络的组播特性方便地提供一些新的增值业务，包括在线直播、网络电视、远程教育、远程医疗、网络电台、实时视频会议等互联网的信息服务领域。

但这里需要注意，组播地址是不分“网络 ID”和“主机 ID”的，也没有子网掩码，只是规定在最高字节中前 4 位分别固定为 1、1、1、0 的 IPv4 地址属于 D 类组播地址（如图 9-7 所示），这样一来整个可用的 D 类组播地址范围为 224.0.0.0～239.255.255.255。



图 9-7 D 类 IPv4 地址的结构

在 D 类组播地址中，除了最高的 4 位是固定的外，其余 28 位均是可变的，所以其 IPv4 地址范围也是很大的。这主要是因为在组播通信中组播 IPv4 地址的应用也非常多，为了更好地规范和管理这些组播 IPv4 地址，根据不同的应用环境和用途又划分为：预留组播地址、公用组播地址、临时组播地址、本地管理组播地址四大类。

（1）预留组播地址。

预留组播地址（也称“永久组播地址”）由 IANA 保留不分配给特定用户使用，仅为公用的组播路由协议分配使用，地址范围为 224.0.0.0～224.0.0.255。使用这些预留组播地址的组播协议包括 RIP、OSPF、IS-IS 等动态路由协议，以及 VRRP（虚拟冗余路由器协议）、IGMP（Internet 组管理协议）、CGMP（Cisco 组管理协议）、IGMP Snooping（IGMP 侦听）和 PIM（协议无关组播）等。

在这个地址组段中，224.0.0.0 不分配，224.0.0.1 分配给本地组播网络所有支持组播的主机，224.0.0.2 分配给本地网络中的所有组播路由器，224.0.0.4 分配给本地网络中的所有 DVMRP 路由器，224.0.0.5 分配给本地网络中的所有 OSPF 路由器，224.0.0.6 分配给本地广播网络中的所有 OSPF 指定路由器（DR），224.0.0.9 分配给本地网络中的所有 RIPv2 路由器，224.0.0.10 分配给本地网络中的所有 IGRP 路由器，224.0.0.13 分配给本地网络中的所有 PIMv2 路由器，224.0.0.22 分配给本地网络中的所有 IGMPv3 路由器。

（2）公用组播地址。

公用组播地址就是可以在全球范围内直接在互联网中使用的组播地址，就像前面所介绍的公网单播 IPv4 地址一样。公网组播地址的范围为 224.0.1.0～224.0.1.255，也是由 IANA 为提出申请并付费的用户进行分配的。

（3）临时组播地址。

临时组播地址就是由企业用户在本企业局域网内部使用的组播地址，地址范围为 224.0.2.0～238.255.255.255，仅在本地局域网内有效，就像前面介绍的私网 IPv4 地址一样。

（4）本地管理组播地址。

本地管理组播地址也是保留使用的，专用于局域网内部组播测试，地址范围为 239.0.0.0～239.255.255.255，仅在特定的本地网络范围内有效。

【注意】在进行组播通信时，在数据链路层“目的 MAC 地址”字段封装的也是组播 MAC 地址。IANA 把 01:00:5E 开头的以太网 MAC 地址块作为组播 IPv4 地址对应的二层组播 MAC 地址。组播 MAC 地址的范围为 01:00:5E:00:00:00～01:00:5E:7F:FF:FF（前 24 位为 MAC 头，固定不变，第 25 位为 0），并要求将 IPv4 组播地址的后 28 位（因为最高的 4 位是固定不变的）映射到 48 位的 MAC 地址空间中。

具体的映射方法是将组播 IPv4 地址中的低 23 位放入 MAC 地址的低 23 位，如图 9-8 所示。至于为什么要映射后面的 23 位，原因就在于根据 IANA 给出组播 MAC 地址段是前 3 个字节（也就是 24 位）来标识单位或者厂商，只有后面 24 位来和 IPv4 地址映射；而给定的地址空间后 3 个字节的最高位相同，都为 0，那么给定的 MAC 地址段内只有 23 位了，所以最终只能把 IPv4 地址

的 28 位丢弃前 5 位，剩下的 23 位和 MAC 的 23 位相映射。注意，这个映射是无需手动进行的，在路由器启动组播协议，站点加入到组播组后就会自动生成。

由于 IPv4 多播地址的后 28 位中只有 23 位被映射到 MAC 地址，这样会有 32 个（IPv4 多播地址中有 5 位是可变的，$=2^5$）IP 多播地址映射到同一 MAC 地址上。

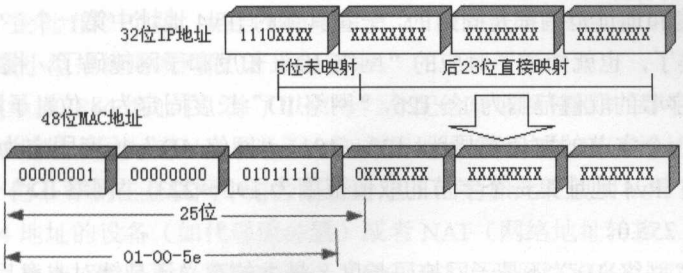


图 9-8 多播 IP 地址到 MAC 地址的映射

5. E 类 IPv4 地址

E 类 IPv4 地址属于 IANA 保留使用，不分配给用户使用的 IPv4 地址，地址段范围为 240.0.0.0～247.255.255.255，其特征是最高 5 位分别是 1、1、1、1、0，如图 9-9 所示，也就是有 27 位是可变的。

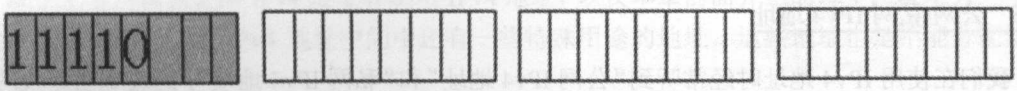


图 9-9 E 类 IPv4 地址的结构



从以上分析可以得出，可以分配给主机和设备接口使用的 IPv4 地址必须是在 A、B、C 类单播地址范围之内，第一个字节的取值范围为 1～223，另外还要除掉第一个字节为 127 的所有 IPv4 地址，以及每个网段中的第一个 IPv4 地址（网络地址）和最后一个 IPv4 地址（广播地址）。组播 IPv4 地址的第一个字节的取值范围为 224～239。

9.1.4 IPv4 地址前缀表示形式

传统的 IPv4 地址表示方法是在给出具体的 IPv4 地址的同时给出它所对应的子网掩码，如 192.168.1.10，255.255.255.0 等。这也是我们在为节点配置 IP 地址的同时要配置子网掩码的原因。但在日常书写时，这样显然比较麻烦、比较长。

为了能更加简便地书写，采取了一种比较简单的地址前缀表示形式，就是在一个 IPv4 地址（可以是网络地址，也可以是主机地址）后面先加上一个斜杠（/），然后在这个斜杠后面直接写出该地址所在网络的“网络 ID”或者“子网掩码”长度，即 IPv4 地址/前缀。

如 192.168.1.10/24 代表的是一个标准的 C 类网络 IPv4 地址，最后的“24”表示该 IP 地址所在网络的网络 ID 长度是 24 位，也就对应它的子网掩码为 255.255.255.0，而 10.1.0.10/8 则代表了一个标准的 A 类网络 IPv4 地址，最后的“8”表示该 IP 地址所在网络的网络 ID 长度是 8 位，也就对应它的子网掩码为 255.0.0.0。

当然，这种表示方式主要还不是为了标识标准的有类网络，因为标准的有类网络中，各类网络的“网络 ID”长度和地址范围都是固定的，一看具体的 IPv4 地址中第一个字节的值就知道属于 A、B、C 中的哪一类了，也就知道了对应的“网络 ID”长度和子网掩码了，根本不用标识。如 A 类 IPv4 地址第一个字节的取值范围为 1～126，“网络 ID”长度固定为 8 位，子网掩码为 255.0.0.0；B 类 IPv4 地址第一个字节的取值范围为 128～191，“网络 ID”长度固定为 16 位，子网掩码为 255.255.0.0；C 类 IPv4 地址第一个字节的取值范围为 192～223，“网络 ID”长度固定为 24 位，子网掩码为 255.255.255.0。

其实无论是“网络 ID”还是子网掩码长度，最大的意义还是针对本章后面将要介绍的无类网络（不按 A、B、C 类进行划分的网络），因为这时它们是可变的，无论第一个字节的值是在哪个取值范围，都可能有各种“网络 ID”、子网掩码长度。如有 1.1.10.20/26 这样的地址，也可以有 192.168.1.10/8 这样的地址，前者是通过子网划分得到，而后者可以通过子网聚合得到。有关子网划分和子网聚合的方法将在本章后面介绍。

9.1.5 公网/私网 IPv4 地址

我们在使用 IPv4 地址时经常听到“公网 IPv4 地址”和“私网 IPv4 地址”，到底哪些是公网 IPv4 地址，哪些是私网 IPv4 地址呢？其实这主要是针对上节所介绍的 A、B、C 这 3 类单播 IPv4 地址而言的，尽管上节介绍到在 D 类组播地址中也有公网与私网之分。下面我们先来介绍私网 IPv4 地址，因为在 A、B、C 类 IPv4 地址中，除了私网 IPv4 地址就是公网 IPv4 地址了。

为了提高 IPv4 地址的重复利用率，在设计 IPv4 地址时就专门在前面介绍的 A、B、C 这 3 类 IPv4 地址中各自划分了一段专门用于各组织局域网内部的地址段，这就是我们前面所说的“私网 IP 地址”或者“局域网专用 IP 地址”或者“专用网络地址”。私网 IPv4 地址在不同公司内部的局域网中是可以重复使用且无需向 IP 地址管理机构申请、注册和购买的。

在 A、B、C 类地址中各自划分的局域网专用地址段如下：

（1）10.0.0.0/8（10.0.0.0，255.0.0.0）。

在 A 类 IPv4 地址中，局域网内部可以使用 10.0.0.0～10.255.255.255 范围内的有效单播 IPv4 地址。如果用地址前缀表示方式（将在本章后面介绍），则为 10.0.0.0/8。在这样一个地址空间中有 24 个“主机 ID”位，相当于最多可以有 2^{24}（16777216）个 IP 地址（包括了网络地址和广播地址），满足了几乎所有大型规模局域网的 IP 地址需求。

（2）172.16.0.0/12（172.16.0.0，255.240.0.0）。

在 B 类 IPv4 地址中，局域网内部可以使用 172.16.0.0～172.31.255.255 范围内的有效 IPv4 单播地址。如果用地址前缀表示方式，则为 172.16.0.0/12。针对具体的网络，仍是 B 类网络中所限制的

16 位“主机 ID”个数，相当于最多可有 2^{16}（65536）个 IP 地址（包括了网络地址和广播地址），满足了几乎所有中等规模局域网的 IP 地址需求。

（3）192.168.0.0/16（192.168.0.0，255.255.0.0）。

对于最为多数的小型局域网，可以使用 C 类 IPv4 地址中 192.168.0.0～192.168.255.255 范围内的有效 IPv4 单播地址。如果用地址前缀表示，则为 192.168.0.0/16。针对具体的网络，仍是 C 类网络中所限制的 8 位“主机 ID”个数，相当于最多可有 2^8（256）个 IP 地址（包括网络地址和广播地址），满足了大多数小型规模局域网的 IP 地址需求。

在 A、B 和 C 类单播 IPv4 地址中，除了以上专门分配给局域网使用的私网 IPv4 地址段外，其他的全部是公网 IPv4 地址。因为 IANA 永远不会为专用地址空间内的 IPv4 地址注册，所以不能在互联网节点上使用，也不可能在互联网中进行路由。也正因为如此，使用私网 IPv4 地址的主机必须与分配了公网 IPv4 地址的设备（如代理服务器）或者 NAT（网络地址转换）路由器将私网 IPv4 地址转换成公网 IPv4 地址，然后再与互联网连接。有关三大主要网络设备品牌的 NAT 技术请大家参见《Cisco 路由器配置与管理完全手册》（第二版）《H3C 路由器配置与管理完全手册》（第二版）和《华为路由器学习指南》。

9.1.6 同种特殊的 IPv4 地址

除了上节所说的公网 IPv4 地址和私网 IPv4 地址，以及本章前面介绍的每个网段均有一个网络地址和广播地址外，在 IPv4 地址空间中还有一些特殊用途的地址，这些地址也是不能分配给主机或设备节点使用的。

1. 自动专用 IPv4 地址

在 Windows 操作系统中，如果你采用的是 DHCP 服务器自动 IPv4 地址分配方式，但本地网络中又没有可用的 DHCP 服务器时，Windows 系统会为主机自动分配一个以 169 开头的 IPv4 地址。这就是所谓的“自动专用 IP 地址”，其地址范围包括一个 B 类地址段——169.254.0.0/16，子网掩码为 255.255.0.0。

这个地址段的 IPv4 地址虽然可以由 Windows 系统自动分配，但是它不能与网络连接，仅用于本地链路间建立邻居关系，就相当于本章后面将要介绍的 IPv6 地址中的“链路本地地址”。

2. 127.0.0.0/8 地址

我们在配置网络设备或者进行系统主机测试时经常用到一个 127.0.0.1 之类的地址（其实有一整个地址段 127.0.0.0/8），称为“环回地址”（Loopback Address）。它不能分配给主机用，主要用于网络软件测试以及本地机进程间通信，在 IP 网络中就是用来测试主机 TCP/IP 协议栈是否工作正常。无论什么程序，一旦使用回送地址发送数据，协议软件立即返回，不进行任何网络传输，所以执行对这类地址的 Ping 操作也只在本机上进行环路测试，用来检测网卡或接口是否工作正常。

在我们没有为主机配置 IPv4 地址时，其实系统会默认分配这样一个地址，虽然不能用于网络连接，但可代表本地主机。

3. 0.0.0.0 地址

严格说来，0.0.0.0 不是一个真正意义上的 IP 地址，因为我们在介绍 IPv4 地址的分类时就已经说到，第一个字节的值不能为 0。但在实际的设备配置和网络中确实又用到这样一个 IPv4 地址。其实它不是特指某个 IPv4 地址，在不同的情形中使用它有不同的含义。

如我们在配置默认路由时就使用了 0.0.0.0 这个 IPv4 地址作为目的网络地址，它是代表所有不清楚路由的目的主机和目的网络。这里的“不清楚”是指在本设备的路由表里没有特定路由表项指明如何到达这些主机或网络。在 DHCP 服务器 IP 地址自动分配服务中，在客户端未分配 IPv4 地址前，客户端在发送 DHCP 报文时在“源 IP 地址”中所填上的就是 0.0.0.0，它代表未知，即可以是由 DHCP 服务器分配的任意 IPv4 地址。有关 DHCP 服务工作原理将在第 11 章中介绍。

9.2 IPv4 子网划分与聚合

本章前面介绍的都是基于标准的 A、B 和 C 类的 IPv4 地址划分方式，但随着计算机网络的普及和互联网应用的高速发展，很快就发现在这种原始的划分方式下，公网 IPv4 地址很快就要耗尽了。为此，就诞生了两种非常重要的技术，那就是 VLSM（Variable Length Subnet Masking，可变长子网掩码）和 CIDR（Classless Inter-Domain Routing，无类别域间路由），把传统标准的 IPv4 有类网络演变成一个更为高效更为实用的无类网络（不再按 A、B、C 这样分类了）。

VLSM 用于 IPv4 子网的划分，也就是把一个大的网络划分成多个小的子网；而 CIDR 则用于 IPv4 子网的聚合，可以把多个连续子网在路由角度汇总成一个更大的子网或者对应的标准网络，这样可以在实现各子网间路由的基础上大大减少路由器中的路由条目，提高路由效率。下面先来简单介绍什么是有类网络，什么是无类网络。

9.2.1 IPv4 子网划分的意义

在正式介绍子网划分的意义之前，先来理清两个通俗的叫法：“有类网络”和“无类网络”。

“有类网络”就是按照 9.1.3 节那样的划分方式把整个单播 IPv4 地址空间划分成 A、B、C 三类，也称“标准网络”或者“自然网段”“主网络”等。每一类均有一些最高比特位的值是固定了的，同一类网络中的“网络 ID”长度也是固定的，即同类网络的子网掩码是固定且一样的，根本不用计算。这样一来，凡是第一个字节的值在 1～126 之间的网络都是 A 类网络，子网掩码都是 255.0.0.0；凡是第一个字节的值在 128～191 之间的网络都是 B 类网络，子网掩码都是 255.255.0.0；凡是第一个字节的值在 192～223 之间的网络都是 C 类网络，子网掩码都是 255.255.255.0。

而所谓“无类网络”则是对“有类网络”进行子网划分，就是把一个大的标准网络划分成多个小的子网，而子网下面又可以根据需要进一步划分。其目的就是把一个大的网络划分成多个小的网络，这样就可以提高 IPv4 地址的利用率，当然这主要是针对 ISP 所使用的公网 IPv4 地址，因为私网 IPv4 地址不同公司可以重复使用，绝大多数企业都根本用不完这么多的私有 IPv4 地址。

划分子网所用的技术就是前面介绍到的 VLSM（可变长子网掩码）。从其名称上可以看出，通

过 VLSM 划分出的子网的子网掩码是可变的，不再像 A、B、C 类 IPv4 网络那样子网掩码是固定的，其子网掩码长度也不再是有类网络中只有 8、16、24 这 3 种整字节长度了，可以是大于等于 8，小于等于 32 的任何一个值，如 12、13、14、25、26、28 等。

这时你可能马上会问，通过 VLSM 划分子网能给我们带来什么好处呢？主要有 3 个方面：减少 IPv4 地址的浪费，提高 IPv4 地址的利用率；减小广播报文的影响，优化网络性能；提高网络安全性，增加安全策略设置灵活性。

（1）减少 IPv4 地址的浪费，提高 IPv4 地址的利用率。

举例来说，先说在局域网中。假设你公司的局域网用户数约为 1000 个左右，也就至少需要 1000 个左右的 IPv4 地址。这时如果想把所有设备都放在一个网段中，这样各设备可以直接进行二层通信，无需三层网络设备。根据表 9-1 给出的三类网络类型所对应拥有的 IPv4 地址数可以知道，你的最佳选择就是用一个 B 类地址网络，但它每个网络可以使用的 IPv4 地址总数达到了 65534 个，相对你只有 1000 个节点的网络来说显然是大材小用了，必将造成有绝大部分的 IPv4 地址在你的网络中根本用不上，白白空置了。这对于小型网络同样如此。假设你们公司想要为每个部门单独配置一个网段，而大多数部门也就是十几二十个人，这样即使各用一个 C 类 IPv4 网段也将使每个部门的网络中有大量地址是用不到的，浪费了。

或许你会说，这没什么问题啊，反正局域网 IP 地址不同公司是可以重复使用的，现有的网络地址数量一个公司无论如何也用不完，而且又不用付费。的确是这样的，所以说子网划分主要是针对公网 IPv4 地址，因为公网 IPv4 地址是不能重复使用的，用一个少一个。如果采用了 VLSM 就可以更加充分地利用每一个公网 IPv4 地址。但它在局域网中仍得到广播应用，因为通过 VLSM 划分子网的意义不仅在于提高 IPv4 地址的利用率这一个方面。

或者你又有疑问了，因为每多划分一个子网就多了两个 IPv4 地址（子网网络地址和广播地址）不能分配给主机和节点使用，这不更浪费了吗？这就要看哪种情况浪费更严重了，多划分一个子网虽然会增加两个 IPv4 地址不能分配给主机和节点使用，但是与它节省的 IPv4 地址数相比就小许多了。况且在公网中还可以是 32 位掩码的单 IP 地址网段，这样每个 IPv4 地址就都可以使用了，更加提高了公网 IPv4 地址的利用率。

（2）减小广播报文的影响，优化网络性能。

广播报文在广播类型的网络（如以太网）中是必不可少的，如要通过 ARP 解析目的主机 MAC 地址时就必须发送广播类型的 ARP 请求报文。由于广播报文一旦发出会向网络中的所有节点发送，因此消耗了大量网络带宽和设备资源。而且网络越大，广播域就越大，不仅一次发送的广播报文副本越多，而且网络中发送的广播报文也越多，这样在网络中的广播流量也就越大，甚至可能出现广播风暴（即大量广播报文占满了整个网络带宽，影响了正常的数据通信）。而划分子网后，网络更小了，广播域也更小了，这样广播报文对用户正常通信的影响也就更小了，产生广播风暴的可能性也就更小了。

如图 9-10 所示是一个 C 类网络，只要其中有一个主机发送广播报文，则网络中的其他 253 个用户主机都会收到这个广播报文。但事实上最终只有一台主机是需要并且接收的，相当于对于另外

252 个用户来说这个广播报文是没用的，但发给这 252 个用户的广播副本仍会在网络中传播，占用了网络中宝贵的带宽资源。这还只是一个用户发送了广播报文，如果网络中同时有多个用户发送广播报文，那么网络中传输的广播流量就更大了，对网络正常使用的影响也就更大了。

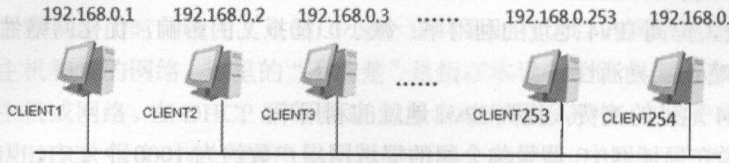


图 9-10　不划分子网的有类网络示例

如果把这个 C 类网络划分成两个子网（如图 9-11 所示），则每个子网中的主机数就减少了一半（划分子网数越多，每个子网中的主机数越少），每个广播报文在传输时要复制的副本数量也就减少了一半，子网中同时发送广播报文的概率也就减小了。

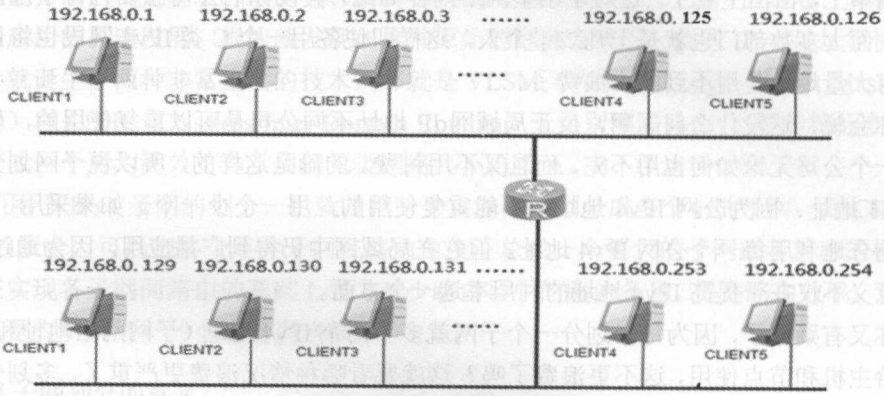


图 9-11　划分子网示例

（3）提高网络安全性，增加安全策略设置灵活性。

这也主要是针对公网环境。我们知道，公网 IP 地址是在互联网中全球用户可以直接访问的（当然这要求用户已在公网中使用这个地址），除非做了特别的限制。而在有类网络中，每个网络的路由是相同的，如 129.168.129.0/24 这个网段的路由条目中就包括这个网络地址，凡是在这个网络中的地址均可通过这条路由到达。现假设 A 公司购买了一个 221.126.128.10～221.126.128.20 的 C 类公网地址段，而 B 公司购买了一个 221.126.128.30～221.126.128.50 的 C 类公网地址段，它们所购买的这些公网地址均用于一个为互联网用户提供服务的小服务器网络中。

如果没有 VLSM，A 公司和 B 公司的局域网均只能通过配置 221.126.128.0/24 这条路由来实现内部局域网与服务器网络的互通。但这样一来，如果没有其他限制措施，则两家公司的这条路由都可能到达对方的服务器，这显然不行，很不安全。但如果每个用户分配整个有类网络的地址段，一方面用户不需要，也不可能花那么大资金来购买用不完的公网 IPv4 地址，另一方面 ISP 也不可能

这样来分配，因为事实上 IPv4 地址资源非常有限，到了有钱还买不到的地步。如果采用 VSLM，则可以很好地根据用户实际需要为每个用户设置一个单独的子网，可以大大降低不同公司中公网地址的不安全性。同时这也方便针对不同子网用户配置不同的用户访问和安全策略，如 QoS 策略、ACL 策略、路由策略等。

9.2.2 VLSM 子网划分的基本思想

通过 VLSM 实现子网划分的基本思想很简单，就是把原来的有类网络 IPv4 地址中的“网络 ID”部分向“主机 ID”部分借位，把一部分原来属于“主机 ID”部分的位变成“网络 ID”的一部分（通常称为“子网 ID”），如图 9-12 所示。这样，原来的“网络 ID”+“子网 ID”=新“网络 ID”，原来的“主机 ID”-“子网 ID”=新“主机 ID”。“子网 ID”长度决定了可以划分子网的数量，等于 2^n，n 为子网 ID 长度。

如向“主机 ID”借 1 位可划分成 2 个子网，借 2 位则可划分成 4 个子网，借 3 位可划分成 8 个子网，依此类推。不能划分成其他数量的子网，如不能把一个网络划分成 3、5、6、7、9 之类个数的子网。但因为由同一个有类网络划分出来的每个子网的“子网 ID”长度一样，这就决定了所划分出来的各子网新的“网络 ID”和“主机 ID”长度也都一样，所以各子网的子网掩码和所包括的 IPv4 地址数都一样，即各子网的 IPv4 地址是对原来有类网络的 IPv4 数除以所划分的子网数的等分值。如一个 C 类网络，向“主机 ID”借 3 位后就划分了 8 个子网，每个子网的 IPv4 地址数=256÷8=32。其实也可以这样简单地计算各子网的 IPv4 地址数，那就是 2 的新主机 ID 位数次方。如前面的示例中，向 C 类网络的“主机 ID”借 3 位后，新的主机 ID 为 8-3=5，这样很快就可算出各子网的 IPv4 地址数是 2^5=32。

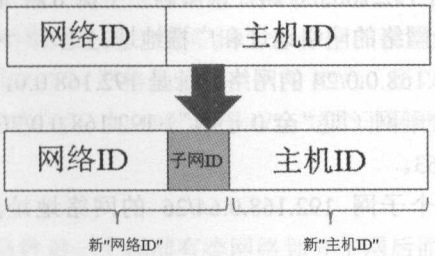


图 9-12 VLSM 子网划分示意图

这样一来，通过 VLSM 可以非常灵活地依据实际所需的地址数来调整所划分的子网大小。如原来一个标准的 C 类网络中有 254 个可以分配给主机或节点使用的 IPv4 地址，现在公司中有 6 个部门，每个部门的人数不超过 30 人，现在想为每个部门分别配置一个子网。如果没有 VLSM，则需要用 6 个 C 类有类网络，而有了 VLSM 后，仅用一个 C 类网络就可以划分出所需的 6 个子网（向“主机 ID”借 3 位，共划分 8 个子网，每个子网 30 个可用的主机 IPv4 地址），而且每个子网中可用的 IPv4 地址更贴近各部门的实际需求，这样就大大提高了 IPv4 地址的利用率。



VLSM 只能划分相同大小的子网，也就是一个网络划分了子网后，各子网的 IPv4 地址数是相同的。要想使各子网大小不一样，则必须同时结合后面将要介绍的 CIDR 技术把其中一些子网再聚合成一些稍大些的子网。另外，子网还可以进一步划分成更小的子网，这就是多级子网划分。具体划分方法将在下节通过具体的示例进行介绍。

9.2.3 “全 0 子网”与“全 1 子网”

“全 0 子网”和“全 1 子网”是针对进行子网划分后“子网 ID”部分的取值而言的，“全 0 子网”代表的是对应子网的“子网 ID”部分各位都是 0，是第一个子网，而“全 1 子网”代表的是对应子网的“子网 ID”部分各位都是 1，是最后一个子网。

按照 RFC950 参考规定，划分子网后只有 n-2 个可用的子网（n 表示总的子网数），第一个子网（也就是“全 0 子网”）和最后一个子网（也就是“全 1 子网”）不可用。那是因为“全 0 子网”与没有划分子网前的标准网络的“网络地址”是一样的，而“全 1 子网”与没有划分子网前的标准网络的“广播地址”是一样的，会造成冲突，到时无法区分到底是代表原来的标准网络还是代表对应的子网了。但是在后来的 RFC1878 规定中，该项规定已被废止了，现在的设备基本上都普遍支持 RFC1878，所以现在来说“全 0 子网”和“全 1 子网”事实上都是可以用的。有些比较老的设备为此还专门配备了一些命令来支持，如在 Cisco 路由器中可以通过 ip subnet-zero 命令配置来实现“全 0 子网”可用。

现在以一个示例来说明为什么在 RFC950 中要规定全 0 子网和全 1 子网不能用。

假设我们有一个 C 类网络 192.168.0.0/24，采用向“主机 ID”借 2 位的方法来划分 4 个子网。下面来分析一下划分前后各个网络的网络地址和广播地址。

- 没划分子网前的 192.168.0.0/24 的网络地址是 192.168.0.0，广播地址是 192.168.0.255。
- 划分子网后的第一个子网（即“全 0 子网”）192.168.0.0/26 的网络地址是 192.168.0.0，广播地址是 192.168.0.63。
- 划分子网后的第二个子网 192.168.0.64/26 的网络地址是 192.168.0.64，广播地址是 192.168.0.127。
- 划分子网后的第三个子网 192.168.0.128/26 的网络地址是 192.168.0.128，广播地址是 192.168.0.191。
- 划分子网后的第四个子网（即“全 1 子网”）192.168.0.192/26 的网络地址是 192.168.0.192，广播地址是 192.168.0.255。

可以看出，子网划分后的第一个子网 192.168.0.0/26 的网络地址和没有划分子网时的 192.168.0.0/24 的“网络地址”是一样的（不考虑子网掩码），都为 192.168.0.0。而划分子网后的最后一个子网 192.168.0.192/26 的“广播地址”和没有划分子网时的 192.168.0.0/24 的“广播地址”也是一样的（也不考虑子网掩码），都为 192.168.0.255。这在早期的一些路由协议（如 RIPv1）中

是容易造成混淆的，因为这些路由协议的路由信息通告中只标明了网络地址，不带子网掩码，而IPv4 报文中也是只有 IPv4 地址信息，没有子网掩码的。但是在后来新版本的路由协议（如 RIPv2、OSPF、IS-IS 等）中的路由信息通告中是带有子网掩码的，这样就可以根据子网掩码来对前面看似一样的“网络地址”和“广播地址”进行区分了。

9.2.4 广播地址的分类

本来广播地址就一种，即每个标准有类网络中所说的最后那个 IPv4 地址，它可以在整个标准网络内进行广播。但自从有了无类网络，进行了子网划分后，根据其广播范围的大小，IPv4 广播地址又分成了以下几种：

（1）网络广播地址。

“网络广播地址”是传统意义上的广播地址，也就是标准的有类网络中的广播地址。网络广播地址可以将数据报文广播发送到本地有类网络内部的所有节点上。IPv4 路由器不转发目的 IP 地址为网络广播地址的广播数据报文，也就是网络广播报文只能在一个本地有类网络内部广播，而不能被路由到其他网络中。

网络广播地址是通过将有类 IPv4 地址（A、B、C 三类）中的所有“主机 ID”部分全部设置为 1 得到的，也就是每个“主机 ID”的八位组均为 255。例如，假设 151.110.0.0 是一个有类别地址的 B 类地址，则其网络广播地址为 151.110.255.255。

（2）子网广播地址。

“子网广播地址”是针对具体子网的广播地址，它仅可以将数据报文发送到相应无类别子网内部的所有节点上。IPv4 路由器也不转发目的 IP 地址为子网广播地址的广播数据报文，也就是子网广播报文只能在一个子网内部广播，而不能被路由到其他子网中。

子网广播地址是通过将无类别地址的“主机 ID”部分全部设置为 1 得到的，因为在包含“主机 ID”的八位组中还包括“网络 ID”（严格地讲是“子网 ID”），所以这时的广播地址中“主机 ID”所对应的八位组值不一定就是 255 了。如 192.168.1.0/26 这个子网的广播地址是 192.168.1.63 而不是 192.168.1.255。

（3）全子网定向广播地址。

“全子网定向广播地址”是针对一个标准有类网络划分子网后而言的，等于这个标准网络的广播地址。它可以将数据报文以广播方式发送到由该标准网络划分的所有子网的节点上。如前面所说的 192.168.1.0/24 标准有类网络通过向“主机 ID”借 2 位的方法划分成：192.168.1.0/26、192.168.1.64/26、192.168.1.128/26、192.168.1.192/26 这 4 个子网，则原来 192.168.1.0/24 有类网络的广播地址 192.168.1.255 就是这 4 个子网的全子网定向广播地址。以这个广播地址作为目 IPv4 地址的广播报文可以在以它划分的所有子网中广播，也就是 IPv4 路由器可以转发这类广播报文。

（4）有限广播地址。

“有限广播地址”是通过将 IPv4 地址的 32 个位全部设置为 1（255.255.255.255）而形成的。在本地“网络 ID”未知的情况下（如采用 DHCP 服务自动分配 IP 地址的客户端），可以使用有限

广播地址来进行本地网络或子网内部所有节点的传送。

IPv4 节点通常仅在自动化配置过程（如启动协议（BOOTP）或 DHCP）中使用有限广播地址，如我们将在第 11 章中介绍的 DHCP 服务，DHCP 客户端发出的请求报文和 DHCP 服务器返回的应答报文都是以 255.255.255.255 作为目的 IPv4 地址，因为那时 DHCP 客户端还没有分配到具体的 IP 地址，也就无法确定它在哪个网络，使用哪个网络的广播 IPv4 地址了。

9.2.5 VLSM 子网划分方法

许多读者朋友一谈到子网划分好像就觉得非常难、非常怕，总也搞不清楚。其实子网划分真的很简单，不信请继续看下去。

在子网划分类试题或者具体网络工程计算中不外乎两种情形：一是已知所需子网数或最大主机数求子网掩码；二是已知子网掩码或子网掩码前缀长度求子网地址范围。

1. 情形一：已知要对某网络划分的子网数或同时要求子网的最大主机地址数，求子网掩码、子网地址范围、网络地址和广播地址

这种情形下的计算步骤如下：

（1）根据已知所需的子网数和最大地址数共同确定划分子网后的新“网络 ID”位数和新“主机 ID”位数。

（2）根据新“主机 ID”位数确定子网划分子网后各字节的地址块大小，由此可进一步确定各子网的地址范围、网络地址和广播地址。

（3）原子网掩码中的 255 字节保持不变，对原子网掩码中非 255 的字节分别用“256-对应字节地址块”得出划分子网后的子网掩码。

【示例 1】一公司想在原来 192.168.1.0/24 标准网络的基础上为 6 个部门各分配一个 IP 子网，最小部门人员数为 25 个员工，求划分后的子网掩码及各子网的网络地址和广播地址。

（1）25 个主机地址最少需要占用 5 位，由此可得划分后的子网“网络 ID”部分最多向“主机 ID”部分借 8-5=3 位，而借 3 位可以划分的子网数是 2^3=8，也符合 6 个部门每个都占用一个子网的要求。

（2）由以上得出最终的子网划分方案中的“主机 ID”部分为 5 位，仅在最后一个字节地址块大小为 2^5=32。

（3）原来 192.168.1.0/24 的子网掩码为 255.255.255.0，值为 255 的保持不变，仅需对最后一个字节用“256-地址块”=256-32=224，最后得出划分子网后的子网掩码为 255.255.255.224。

把原来网络中的所有 IPv4 地址按每段 32 个来划分，即可得出每个子网的地址范围（注意，第一个 IPv4 地址是从 0 开始计算）如下：192.168.1.0～192.168.1.31、192.168.1.32～192.168.1.63、192.168.1.64～192.168.1.95、192.168.1.96～192.168.1.127、192.168.1.128～192.168.1.159、192.168.1.160～192.168.1.191、192.168.1.192～192.168.1.223、192.168.1.224～192.168.1.225。

各子网的地址范围知道了，自然就知道它们的网络地址和广播地址了。每个子网的网络地址是对应地址段中的第一个地址，每个子网的广播地址是对应地址段中的最后一个地址。

【示例 2】一公司想把原来使用的 172.16.0.0/8 标准网络分成 12 个子网，其中一个最大的子网节点数在 4000 个左右。求每个子网的子网掩码、地址范围、网络地址和广播地址。

这个示例同样是已知要划分的子网数和最大地址数，具体的划分方法很简单，与上一个示例一样。具体的计算步骤如下：

（1）本示例中要划分成 12 个子网，与 12 最接近的可划分子网数就是 16，由此可见要向原来的“主机 ID”借 4 位（2^4=16），也就是第 3 个八位组的高 4 位。这样一来子网的“主机 ID”位数就仅有 12（16-4）位了，则每个子网可使用的 IPv4 地址数为 4094（2^{12}-2）（同样是因为要减去每个子网中不可分配给节点使用的网络地址和广播地址）个，恰好符合该公司中最大子网地址数 4000 的要求。

（2）知道了新“主机 ID”大小，也就是知道了各子网的地址块大小。本示例中新“主机 ID”共 12 位，总地址块大小为 4096（2^{12}）。但这个地址块是分布在第 3 个字节和第 4 个字节中，第 3 个字节有最低的 4 位，对应的子地址块大小为 16，而第 4 个字节包括了全部的 8 位，对应的子地址块大小为 256。

（3）根据以上得出的两个子地址块大小和上节介绍的公式可以得出子网掩码为：原“网络 ID”(255.255).(256-16).(256-256)，即 255.255.240.0。

（4）知道了总地址块大小，也就是知道了每个子网的地址范围。本示例的地址块大小为 4096，也就是把整个原来 B 类网络中的 65536 个地址按每块 4096 个地址划分成 16 等份，得到的 16 个子网的地址范围如下：172.16.0.0～192.16.15.255（172.16.0.0/20）、172.16.16.0～172.16.31.255（172.16.16.0/20）、172.16.32.0～172.16.47.255（172.16.32.0/20）、172.16.48.0～172.16.63.255（172.16.48.0/20）、172.16.64.0～172.16.79.255（172.16.64.0/20）、172.16.80.0～172.16.95.255（172.16.80.0/20）、172.16.96.0～172.16.111.255（172.16.96.0/20）、172.16.112.0～172.16.127.225（172.16.112.0/20）、172.16.128.0～172.16.143.225（172.16.128.0/20）、172.16.144.0～172.16.159.225（172.16.144.0/20）、172.16.160.0～172.16.175.225（172.16.160.0/20）、172.16.176.0～172.16.191.225（172.16.176.0/20）、172.16.192.0～172.16.207.225（172.16.192.0/20）、172.16.208.0～172.16.223.225（172.16.208.0/20）、172.16.224.0～172.16.239.225（172.16.224.0/20）、172.16.240.0～172.16.255.225（172.16.240.0/20）。

同样，各子网的地址范围知道了，自然也就知道了它们的网络地址和广播地址。然后随便选择其中的 12 个子网使用即可，最好是 12 个连续的子网，这样方便进行路由汇总。同时，如果你的设备不支持全 0 子网和全 1 子网，则不要选择第一个子网和最后一个子网。

【示例 3】每个子网有不超过 58 个节点要分配 IPv4 地址，最合适的子网掩码是（　　）。

A．255.255.255.192　B．255.255.255.248　C．255.255.255.224　D．255.255.255.240

这道题很简单，也是给出了子网中的最大地址数，即 58 个。由此可见，最恰当的方式是对一个 C 类标准网络进行子网划分，因为 58 小于一个字节的地址数 256。而地址块中与 58 最接近的一个就是 64（注意，地址块都是 2^n，n 代表“主机 ID”的位数），即 2^6。这样一来立即可以得出它的子网掩码为 255.255.255.(256-64)，即 255.255.255.192。

2. 情形二：已知子网前缀或子网掩码，求子网地址范围、网络地址和广播地址

这种情形下的计算方法如下：

（1）根据已知子网掩码前缀或子网掩码确定该子网各字节的地址块大小。方法是：对子网掩码中的各非 255 字节用“256-对应字节地址块”，或者用 256 除以通过子网掩码前缀得知划分的子网数，均可得出各非 255 字节所对应的数值。

【说明】只要求第一个非 255 字节的地址块大小，为全 0 的字节地址块大小均为 256。

（2）根据划分子网后的子网掩码中各非 255 字节的地址块大小确定每个子网的地址范围、网络地址和广播地址。

【示例 4】求 172.16.1.0/18 子网中的最大主机 IPv4 地址。

（1）由 172.16.1.0/18 可知其子网掩码前缀为 18，得出其子网掩码为 255.255.192.0（前 18 位全为 1，后 14 位全为 0）。

（2）子网掩码中第一个非 255、非 0 字节是第 3 个字节（从左边数起），用 256-192（此为第 3 个字节的子网掩码值）或者用 256 除以子网数 2^2（第 3 个字节中有 2 位为子网 ID）均可得出该字节的地址块大小为 64。由此可以得出 4 个子网的地址范围为：172.16.0.0～172.16.63.255、172.16.64.0～172.16.127.255、172.16.128.0～172.16.191.255、172.16.192.0～172.16.255.255。

显然 172.16.1.0/18 子网是属于第一个子网，最大可用的主机 IPv4 地址为 172.16.63.254。

【示例 5】求 10.32.0.0，255.224.0.0 子网中最大可分配给主机的 IPv4 地址。

这道题是已知了子网的子网掩码，这时可根据子网掩码 255.224.0.0 很快得出该子网中的“主机 ID”分布在 IPv4 地址后面 3 个字节中，各自的子地址块大小分别是：32（256-224）、256（256-0）、256（256-0）。

在这种“主机 ID”分布在多个字节的情形中，起关键作用的就是包含在“主机 ID”中的第一个字节，如这里 IPv4 地址中第二个字节的“224”，相当于告知了该字节中的子地址块大小就是 32（256-224），由此很快可以得出各子网的该字节取值范围就依次是 0～31，32～63，64～95，…。题中给出的 10.32.0.0 恰好在第二个子网中，它的 IPv4 地址段为 10.32.0.0～10.63.255.255，因为广播地址不能分配给主机使用，所以该子网中可分配给主机使用的最大 IPv4 地址就是 10.63.255.254。

【示例 6】求 202.112.14.137，255.255.255.224 子网的网络地址和广播地址。

这道题也是已知了子网的子网掩码，可根据子网掩码 255.255.255.224 得出其地址块大小为 32（256-224）。因为它是一个 C 类标准网络地址，所以“主机 ID”仅在 IPV4 地址中的第 4 个字节。根据地址块大小 32 可以把第 4 个字节的值划分为：0～31，32～63，64～95，96～127，128～159，…。由此可见 202.112.14.137 是在第 5 个子网中，网络地址是 202.112.14.128，广播地址是 202.112.14.159。

3. 多级子网划分示例

上面介绍的全是单级子网划分的示例，多级子网划分的方法与单级子网划分的方法是一样的，只不过是在子网的基础上进行再次划分。下面也举一个示例。

【示例 7】把 IPv4 子网 10.32.0.0/11 划分成 8 个子网，求重新划分子网后的子网掩码和第 3 个子网的网络地址和广播地址。

这属于上节介绍的第二种情形，已知了地址前缀也就相当于已知了子网掩码。本示例是地址前缀为 11，也就是原子网的子网掩码中前 11 位全为 1，后 21 位全为 0，得到子网掩码为 255.224.0.0。

现要对 10.32.0.0/11 子网进行二次子网划分，且划分的子网数是 8 个，也就是再要向“主机 ID”借 3 位，由此可知新划分的子网中“网络 ID”一共有 14（11+3）位，得到新子网掩码为 255.252.0.0。

得出了新子网掩码后就可以得出属于“主机 ID”部分的字节的子地址块大小（分别用 256 去减即可）。从新子网掩码为 255.252.0.0 可以得出，“主机 ID”分布在第 2 个、第 3 个和第 4 个字节中，各自的子地址块大小分别为 4（256-252）、256（256-0）、256（256-0）。同样关键是第一个属于“主机 ID”的字节的子地址块大小，即 4。所以 10.32.0.0/11 重新划分子网后的第二个字节的值是以 4 为单位进行划分的。从第一个新子网开始，第二个字节的值依次是 32、35、39、43、47、51、55、59、63。题中所求的第 3 个子网就是第二个字节值为 39 的子网，它的地址段范围为 10.39.0.0～10.43.255.255，其网络地址和子网地址自然就分别为 10.39.0.0/14 和 10.43.255.255/14。

9.2.6 CIDR 子网聚合的基本思想

前面具体介绍了通过 VLSM 进行子网划分的方法，本节再来介绍子网聚合的方法。但这里所说的“子网聚合”不是在物理上把原来划分后的子网又重新聚合成一个大的网络，而是计算一条同时用于划分后各子网数据转发的聚合路由（或称“汇总路由”），以便减少设备上的路由表项数量，提高路由表项查找效率。

这对于当前的一些大型企业或者 ISP 骨干网核心路由器设备来说非常必要的，因为这些核心设备连接的网络非常多，如果每一个子网都以具体的明细路由来出现的话，设备路由表中的路由表项数量将非常多，这样在对用户数据进行转发时，路由表项的查找效率会大大降低。

子网聚合所使用的技术是本章前面提到的 CIDR（无类别域间路由）。通过它可以计算出通过 VLSM 划分后的各子网的聚合路由。聚合路由的计算关键也是要计算出聚合后的网络 ID 长度，它的实现方法很简单，可以简单地理解为前面介绍的 VLSM 的逆过程。具体做法是把原来子网中的“主机 ID”向“网络 ID”部分借位（如图 9-13 所示），实现“主机 ID”扩展（可容纳的主机数增多），“网络 ID”部分缩小，最终达到聚合子网路由、精减路由表项的目的。但聚合网络的“网络 ID”位数不能小于 8。

在通过 CIDR 进行子网聚合时需要注意以下几项原则：

（1）与 VLSM 划分中向“主机 ID”借 n（n≥1）位可划分出 2^n 个子网对应，通过 CIDR 向“网络 ID”借 n 位则可聚合 2^n 个子网。也就是“网络 ID”每减少一位，所聚合的子网数就是前面所聚合的子网数的 2 倍。如向“网络 ID”借 1 位时能聚合两个子网，借 2 位时则能聚合 4（2×2）个子网，借 3 位时则能聚合 8（2×4）个子网，依此类推。

（2）与 VLSM 只可以划分出 2^n（如 2、4、8、16 等）个子网，不能划分出非 2^n（如 3、5、6、7、9、10 等）个子网一样，子网聚合也只能聚合 2^n 个子网，不能聚合非 2^n 个子网，尽管你可能实际需要只聚合非 2^n 个子网。例如本来你的网络中只有 192.168.1.64/26、192.168.1.128.0/26、

192.1681.192/26 这 3 个子网，如果想对这 3 个子网进行聚合的话，最终聚合的肯定不会只针对这 3 个子网，而是把你网络中没有的 192.168.1.0/26 这个子网包括进去了。也就是聚合的路由会同样适用于 192.168.1.0/26 这个子网。

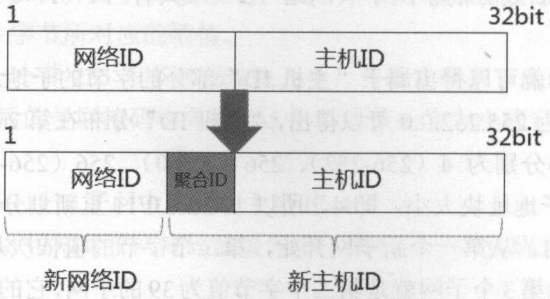


图 9-13　CIDR 子网聚合示意图

（3）被聚合的最后一个子网必须是原网络划分出的第 2^m（m≥1）个子网，如第 2、4、8、16 等个子网，如果不是则最终被聚合的子网是向后延续到最近一个第 2^m 个子网，且必须连续，同时要满足前面所说的被聚合子网数必须是 2^n 个。如对 192.168.1.0/24 这个有类网络划分了 8 个子网，则从第 1 个到第 8 个子网的网络地址分别为：192.168.1.0/27、192.168.1.32/27、192.168.1.64/27、192.168.1.96/27、192.168.1.128/27、192.168.1.160/27、192.168.1.192/27、192.168.1.224/27。

根本以上介绍的原则，192.168.1.0/27、192.168.1.32/27 这两个子网可以聚合，192.168.1.0/27、192.168.1.32/27、192.168.1.64/27、192.168.1.96/27 这 4 个子网可以聚合，192.168.1.128/27、192.168.1.160/27、192.168.1.192/27、192.168.1.224/27 这 4 个子网也可以聚合，当然全部 8 个子网更可以被聚合。但是不能仅对 192.168.1.32/27、192.168.1.64/27 这两个子网进行聚合，因为最后一个被聚合的子网 192.168.1.96/27 是属于原有类网络划分出的第 3（不是 2^m）个子网，不符合本条原则规定。这两个子网最终聚合的结果肯定还会包括 192.168.1.96/27 这个子网，同时还包括最前面的 192.168.1.0/27 子网，因为最终被聚合的子网数要符合 2^n 个。

也不能仅对 192.168.1.32/27、192.168.1.64/27、192.168.1.96/27、192.168.1.128/27 这 4 个子网进行聚合，因为最后一个被聚合的子网 192.168.1.128/27 是原有类网络划分的第 5（也不是 2^m）个子网，也不符合本条规定。这 4 个子网聚合的最后一个子网是 192.168.1.224/27，同时为了满足被聚合的子网数必须是 2^n 个，所以最前面的 192.168.1.0/27 子网也将同时被聚合。同样的道理，也不能仅对 192.168.1.0/27、192.168.1.32/27、192.168.1.64/27、192.168.1.96/27、192.168.1.128/27，或者 192.168.1.32/27、192.168.1.64/27、192.168.1.96/27、192.168.1.128/27、192.168.1.160/27，或者 192.168.1.64/27、192.168.1.96/27、192.168.1.128/27、192.168.1.160/27、192.168.1.192/27，或者 192.168.1.96/27、192.168.1.128/27、192.168.1.160/27、192.168.1.192/27、192.168.1.224/27 这些 5 个子网进行聚合，因为被聚合子网数不是 2^n 个。这些子网聚合的最终结果是什么大家可以自己根据前面的规则进行分析。

9.2.7 子网聚合方法及示例

子网聚合的计算方法有以下两种：

- 直接看最终被聚合的子网数得出，聚合 2 个子网，“网络 ID”长度减 1；聚合 4 个子网，“网络 ID”长度减 2；聚合 8 个子网，“网络 ID”长度减 3，依此类推。
- 求相同的二进制“网络 ID”位数，就是把各子网的“网络地址”用二进制形式表示，然后连续完全相同的部分作为聚合后的“网络 ID”。这是一种比较笨但最有效的方法。

下面通过几个示例来进行具体介绍。

【示例 1】求 192.168.1.0 /27、192.168.1.32 /27、192.168.1.64 /27、192.168.1.96 /27 四个连续子网聚合后的网络地址和子网掩码。

先用方法一来求解。从题中的已知条件很容易看出，这里被聚合的 4 个子网是连续的，符合被聚合子网数必须为 2^n 个的要求；同时最后一个被聚合的子网 192.168.1.96 /27 为原网络 192.168.1.0/24 划分后的第 4 个子网，也符合最后一个被聚合的子网为第 2^m 个的要求。所以直接可以得出，聚合后的“网络 ID”长度为原来的 27 减 2，等于 25，网络地址的计算可通过把其中一个子网网络地址中的前 25 位保持不变，后 7 位全部置 0 得到，最终的网络地址为 192.168.0.0/25，子网掩码为 255.255.255.128.0

下面再通过方法二来计算，也可验证上面通过方法一得出的结果是否正确。把题中的 4 个连续子网的网络地址用二进制形式表示如下：

11000000.10101000.00000000.00000000

11000000.10101000.00010000.00000000

11000000.10101000.00011000.00000000

11000000.10101000.01100000.00000000

然后把这 4 个子网的“网络 ID”中相同的部分（即 11000000.10101000.0）保持不变，其余各位均置 0，即得出聚合后的网络地址为 192.168.0.0/25。这里聚合的是 4 个子网，所以最终的子网掩码长度比原来子网的子网掩码长度 27 小 2，也得到聚合后网络的子网掩码长度为 25。然后把前 25 位全部置 1，后 7 位全部置 0，就得到聚合网络的子网掩码为 255.255.128.0。

通过方法二的计算验证了方法一计算的结果是正确的。

【示例 2】求 192.168.4.0/24、192.168.5.0/24、192.168.6.0/24、192.168.7.0/24 这 4 个有类网络聚合后的网络地址和子网掩码。

先用方法一来求解。这是 4 个有类网络的聚合，因为这里被聚合的最后一个网络是 192.168.7.0/24，也可看成是 192.168.0.0/21 网络划分出来的其中 4 个子网（一共向“网络 ID”借了 3 位，划分出了 8 个子网）。但这里最后一个被聚合的子网是 192.168.7.0/24，恰好是第 8 个子网（第一个子网可以看做是 192.168.0.0/24），符合最后一个被聚合子网为第 2^m 个的要求，所以最终聚合的结果是可以仅针对题中这 4 个给出的网络进行聚合的。

根据前面得出的规律，4 个子网聚合，就只需要向“网络 ID”部分借 2 位，相当于是被聚合

子网中原来的“网络 ID”减 2，即 24-2，等于 22。然后把题中的任意一个子网的网络 ID 前 22 位保持不变，后 10 位全部置 0，就得到聚合后的网络地址为 192.168.1.0/22，聚合后网络的子网掩码为 255.255.252.0。

下面再通过方法二来计算，以验证方法一计算结果的正确性。把题中的这 4 个有类网络的网络地址转换成如下二进制形式：

11000000.10101000.00000100.00000000
11000000.10101000.00000101.00000000
11000000.10101000.00000110.00000000
11000000.10101000.00000111.00000000

结果得出相同部分共有 22 位，把余下的 10 位全部置 0，就得出聚合后的网络地址为 192.168.1.0/22。然后把最高 22 位全部置 1，后 10 位全部置 0，就得出聚合后网络的子网掩码为 255.255.252.0。与前面采用方法一计算的结果完全一样。

这里假设题中的要求是只聚合 192.168.4.0/24、192.168.5.0/24、192.168.6.0/24 这 3 个有类网络，大家计算一下，其结果仍是与上面一样，因为这里最后一个被聚合的有类网络不是原网络划分出来的第 2^m 个，所以不能作为最后一个被聚合的网络，最终聚合的结果仍将包括 192.168.7.0/24 这个网络。

【示例 3】求 192.168.5.0/24、192.168.6.0/24、192.168.7.0/24、192.168.8.0/24 这 4 个有类网络聚合后的网络地址和子网掩码。

先用方法一来求解。这道题看似与上面差不多，实际聚合的结果却大不一样。因为题中最后一个被聚合的网络是 192.168.8.0/24，可以看出是 192.168.0.0/22 网络（因为网络数大于 8，所以至少要向“网络 ID”借 4 位，划分出来 16 个子网）划分出来的第 9 个子网，不符合我们前面介绍的最后一个被聚合的子网必须是第 2^m 个子网的要求。它最终聚合的结果是，最后一个子网为下一个第 2^m 个子网，本示例中就是第 16 个子网 192.168.15.0/24 了。再加上被聚合子网数必须是 2^n 个的要求，所以最终肯定是全部的 16 个子网都要参与聚合（否则 192.168.5.0/24～192.168.15.0/24 只有 11 个子网），所以可以很快得出最终聚合后的网络 ID 是 24-4=20（因为只有向“网络 ID”借 4 位才能划分出 16 个子网）。

将题中任意一个网络地址的前 20 位保持不变，后 12 位全部置 0，就得到聚合后网络的网络地址为 192.168.0.0/20，聚合后网络的子网掩码是前 20 位全为 1，后 12 位全为 0，得到 255.255.240.0。

下面同样也再采用方法二来计算，以验证方法一计算结果的正确性。直接把题中的 4 个有类网络的网络地址转换成如下二进制形式：

11000000.10101000.00000101.00000000
11000000.10101000.00000110.00000000
11000000.10101000.00000111.00000000
11000000.10101000.00001000.00000000

经过比较发现，这 4 个子网只有 20 位是完全相同的，也就是聚合后的子网掩码是 20 位，而不

是上例中所得到的 22 位，尽管它们聚合的都是 4 个连续网络，原因在方法一中已有分析。然后把各有类网络 IPv4 地址中的前 20 位保持不变，后 12 位全部置 0，就得到聚合后网络的网络地址为 192.168.0.0/20，对应的子网掩码为 255.255.240.0，与方法一的计算结果完全一样。

9.2.8 网络地址、广播地址和主机地址的考虑

在一些考试中，有关可用主机 IPv4 地址的计算是一个重点，这涉及到网络地址和广播地址的排除。这在有类网络中是非常容易的，就是除了对应有类网络中的网络地址（第一个 IPv4 地址）和广播地址（最后一个 IPv4 地址）外均是可分配给主机使用的 IPv4 地址，而且网络地址至少最后一个字节为 0，广播地址至少最后一个字节为 255。

在无类网络中就不能按有类网络来区分网络地址、广播地址和主机 IPv4 地址了，因为在无类网络中，“网络 ID”和“主机 ID”都可能不是连续的完整字节，而是可能其中有一个字节既包括了“网络 ID”又包括了“主机 ID”。而且经过子网划分后，原来整个地址段被分成了多段，而每个子网都有一个网络地址和广播地址，所以子网的网络地址可能不再是原来整个网络中的第一个 IPv4 地址，广播地址也可能不是原来整个网络中的最后一个 IPv4 地址。

但无类网络与有类网络有一点是相同的，那就是网络地址是子网的第一个 IPv4 地址，广播地址是子网的最后一个 IPv4 地址。总的来说需要注意以下几个方面：

（1）最后一个字节为 0 的 IPv4 地址不一定是网络地址。

如 172.16.2.0/18 就不是网络地址，它是 172.16.0.0/18 这个子网中的一个中间 IPv4 地址，既不是第一个也不是最后一个。因为 172.16.0.0/18 这个子网的地址范围为 172.16.0.0～172.16.3.255，它的网络地址为 172.16.0.0/18。

（2）最后一个字节为 255 的 IPv4 地址不一定是广播地址。

同样如 172.16.2.255/18 就不是广播地址，它也是 172.16.0.0/18 这个子网中的一个中间 IPv4 地址，既不是第一个也不是最后一个。同样是因为 172.16.0.0/18 这个子网的地址范围为 172.16.0.0～172.16.3.255，它的广播地址为 172.16.3.255/18。

（3）最后一个字节为 0 或 255 的也可能是主机 IPv4 地址。

在有类网络中最后一个字节为 0 或 255 的肯定分别是网络地址或广播地址，是不能作为主机 IPv4 地址的。但在无类网络中，这个又是可能的。如前面说的 172.16.2.0/18、172.16.2.255/18 都是 172.16.0.0/18 这个子网可用的主机 IPv4 地址，其实还有 172.16.1.0/18、172.16.3.0/18、172.16.0.255/18、172.16.1.255/18 都是这个子网可用的主机 IPv4 地址。

（4）网络地址的最后一个字节不一定为 0。

在有类网络中，网络地址的最后一个字节肯定为 0，但在无类网络中网络地址还可能最后一个字节不是 0。如 192.168.1.0/24 这个有类网络通过向“主机 ID”借 2 位划分成 4 个子网，分别为 192.168.1.0/26、192.168.1.64/26、192.168.1.128/26、192.168.1.192/26。后面 3 个子网的网络地址的最后一个字节都不为 0。

（5）广播地址的最后一个字节不一定为255。

同样以上面的192.168.1.0/24这个有类网络划分为4个子网的例子进行介绍，除了最后一个子网192.168.1.192/26的广播地址最后一个字节为255外，前面3个子网的广播地址分别为192.168.1.63/26、192.168.1.127/26、192.168.1.191/26。

9.3 IPv6地址基础

本章前面两节介绍了IPv4协议的编址方案，相比起来，IPv6协议的编址方案与IPv4协议编址方案不仅区别很大，而且要复杂许多。IPv6协议与IPv4协议相比，最大的特征就是地址空间增大了许多倍，因为IPv6地址的位数由原来IPv4地址的32位激增到了128位，相当于IPv4地址空间的4倍。

与IPv4地址通常是采用圆点分隔的十进制数表示类似，IPv6地址是使用由冒号分隔的十六进制数表示。每段4位十六进制数（不区分大小写），相当于16位二进制数，2个字节，如图9-14所示。由于地址空间扩展了3倍，所以从目前来看可以说是彻底解决了IPv4地址缺乏的问题，据悉IPv6地址全部使用后可以使全球每个人平均拥有10多个地址。

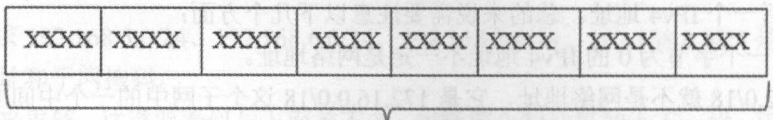


图9-14　128位IPv6地址的格式

另外，与IPv4地址分成“网络ID”和“主机ID”类似，128位的IPv6地址也分成“网络前缀”和“接口标识”两个部分。“网络前缀”对应IPv4地址中的“网络ID”，用于标识IPv6地址所在的网络，“接口标识”对应IPv4地址中的“主机ID”，用于标识具体的主机或节点。

【说明】对于IPv6单播地址来说，如果地址的最高3位不是000，则“接口标识”部分必须为64位；如果地址的最高3位是000（包括IPv4兼容地址和IPv4映射地址），则没有此限制。有关IPv6地址的分类将在本章后面介绍。

9.3.1 IPv6地址表示形式

IPv6地址长度4倍于IPv4地址，一共128位二进制，显然在平时的书写中不能直接用二进制。为了最大限度地缩短IPv6地址的书写长度，采用了较大的十六进制，因为一位十六进制可以代表4位二进制，这样最长也有32位十六进制，每4位的中间用一个冒号分隔。

1. 基本的“冒号十六进制”表示形式

在我们的日常使用中，IPv4地址是每8位二进制以点分十进制形式（每个字节中间以小圆点分隔）来表示的，如192.168.10.12。对于IPv6地址，128位二进制则是按每16位（是IPv4地址

中每段位数的 2 倍）为一组来分隔的，然后转换成 4 个十六进制数（注意不是十进制了），这样就一共划分成了 8 段（也是 IPv4 块数的 2 倍）十六进制数。且相邻的 16 位块是以半角冒号（不再是 IPv4 地址中的小圆点）隔开。所以，生成的 IPv6 地址表示形式叫做“冒号十六进制”表示形式。

下面是一个二进制形式的 IPv6 地址（共 128 位）示例：

0011111111111110:0010100100000000:1101000000000101:0000000000000000:
0000001010101010:0000000011111111:1111111000101000:1001110001011010

在日常的使用中，IPv6 地址不是以二进制来表示的，而是以十六进制表示的（主要也是为了书写方便，如果也用十进制来表示的话，这个十进制数就会很大，每段最大数为 65535）。把每段都转换成十六进制，相邻的块用半角冒号隔开即可。以上示例转换的结果为（注意由 8 段组成）：3FFE:2900:D005:0000:02AA:00FF:FE28:9C5A。

2. 前导零省略表示形式

从上面的十六进制 IPv6 地址来看，仍然相当长，不便于书写，也容易出错。为了进一步简化 IPv6 地址，在 IPv6 地址编址方案中新增加了一个规定，那就是如果在一个由冒号分隔的十六进制段中有前导零则可以省略（中间和后面的 0 不能省略）。如 3FFE:2900:D005:0000:02AA:00FF:FE28:9C5A 中的 02AA 可以写成 2AA，00FF 可以直接写成 FF。如果整个地址段都是 0，则只需要写一个 0 即可，如上面示例中的 0000 就可以写成 0。这样一来，这个示例就可以进一步简化写成：3FFF:2900:D005:0:2AA:FF:FE28:9C5A。

又如 1080:*0000:0000:0000*:0008:0810:213C:123A 这样的 IPv6 地址，采用以上方案后就可直接写成 1080:*0:0:0*:8:810:213C:123A。

3. 压缩零表示形式

虽然通过前面介绍的前导零省略法已对 IPv6 地址表示形式进行了简化，但由于用十六进制表示的 IPv6 地址最长可能有 32 位之多，即使按上述方法去掉了前导零仍可能比较长。为了进一步简化，又导入了双冒号的规则，那就是可以用一个双冒号置换地址中连续值为“0”的十六进制段，这就是通常所说的“双冒号置换法”。

如上面的 1080:*0:0:0*:8:810:213C:123A 地址中间有 3 个连续的地址段为“0”，此时可以对这 3 个连续为的 0 地址段用一个冒号（:）置换（相当于直接把连续 0 及它们之间原来的冒号一起删除）。这样一来，上面示例就可以进一步简写成 1080::8:810:213C:123A，更加简化了。

同样，也可以将 FE80:*0:0:0*:2AA:FF:FE9A:4CA2 的 IPv6 地址压缩成 FE80::2AA:FF:FE9A:4CA2；可以将 FF02:*0:0:0:0:0:0*:2 的 IPv6 地址压缩成 FF02::2；可以将 2000:*0:0:0:0:0:0*:1 的 IPv6 地址表示为 2000::1。0:0:0:0:0:0:0:1 等价于::1；0:0:0:0:0:0:0:0 等价于 ::。

【注意】“双冒号置换法”中的双冒号所代表的值为 0 的连续 16 位地址段个数不限（至少为 2），但是双冒号在一个地址中只能使用一次，也就是说不能重复使用它来替换多个分隔的连续地址段中的 0。例如有这样一个 IPv6 地址 *0:0:0:*AB98:123C:*0:0:0*，按上述双冒号置换原则可以缩写成*::*AB98:123C:0:0:0 或 0:0:0:AB98:123C*::*，但却不能写成*::*BA98:7654*::*，因为这样在一个地址中两次使用冒号置换后，你就无法确定每个“::”代表多少位 0 了。

另外，“双冒号置换法”中的双冒号只能置换连续的值为 0 的十六进制地址段，不能置换非连续的值为 0 的多个地址段。如 FE80:0:2AA:0:FF:0:FE9A:4CA2 就不能用双冒号置换了。也不能对非全 0 的十六进制地址段中的部分 0 进行替换，如不能将 FF02:30:0:0:0:0:0:5 表示为 FF02:3::5，因为在这个书写格式中把作为地址段“30”中一部分的“0”也省略了。

最后总结一下 IPv6 地址的书写规则：

- 每个十六进制地址段中的前导“0”都可以省略，如果该段全为 0，可以只写一个“0”。
- 当 IPv6 地址中有多个连续的值为“0”的十六进制地址段时，可以用一个双冒号转换这些连续的“0”，但双冒号在一个 IPv6 地址中只能使用一次。
- 不能用双冒号转换十六进制地址段中一部分的“0”，即使是地址段中最后一个十六进制 0。

9.3.2 IPv6 地址中的二进制与十六进制转换

因为在 IPv6 地址中使用的是二进制或者十六进制，而不再是 IPv4 地址中的二进制或者十进制，所以这就涉及到了二进制与十六进制之间的转换问题。这在第 1 章中已有全面的介绍，在此仅就 IPv6 地址中的二进制与十六进制转换方法作一下简单介绍。其实就是给出了 4 位二进制所对应的十六进制，如表 9-2 所示（其中的十进制主要是为了方便对照）。

表 9-2 十进制、十六进制和二进制之间的转换对照表

十进制	十六进制	二进制	十进制	十六进制	二进制
0	0	0	8	8	1000
1	1	1	9	9	1001
2	2	10	10	A	1010
3	3	11	11	B	1011
4	4	100	12	C	1100
5	5	101	13	D	1101
6	6	110	14	E	1110
7	7	111	15	F	1111

要将十六进制数转换成二进制数，应将每个十六进制数字转换成对等的 4 位二进制数字。例如，要将十六进制数 03D8 转换成二进制数，应将每个十六进制数字（0、3、D 和 8）分别转换成二进制数字。这样，0x03D8 就是 0000 0011 1101 1000 或 0000001111011000。

要将二进制数转换成十六进制数，应从低序位开始将二进制数分割成包含 4 位的块，最后不足 4 位时在前面以“0”补足 4 位。然后，再将每个包含 4 位的块转换成与其相等的十六进制形式。例如，要将二进制数 110000110101110 转换成十六进制数，应先将整个数分割成 4 位一块的数字块，即 0110 0001 1010 1110（第一个 0 是补的），然后将每块转换成十六进制数字，即 61AE。

9.3.3 IPv6 地址类型

IPv6 协议主要定义了 3 种地址类型：单播地址（Unicast Address）、组播地址（Multicast Address）和任播地址（Anycast Address）。与原来的 IPv4 地址相比，新增了“任播地址”类型，取消了原来 IPv4 地址中的“广播地址”，因为在 IPv6 中广播功能是通过组播来完成的。

- 单播地址：用来唯一标识一个接口，类似于 IPv4 地址中的单播地址。发送到单播地址的数据报文将被传送给此地址所标识的一个接口。
- 组播地址：用来标识一组接口（通常这组接口属于不同的结点），类似于 IPv4 地址中的组播地址。发送到组播地址的数据报文被传送给此地址所标识的所有接口。
- 任播地址：用来标识一组接口（通常这组接口属于不同的结点）。发送到任播地址的数据报文被传送给此地址所标识的一组接口中距离源结点最近（根据使用的路由协议进行度量）的一个接口。

IPv6 地址类型由“地址前缀”部分来确定，主要地址类型与地址前缀的对应关系如表 9-3 所示。有关 IPv6 地址前缀将在本章后面介绍。

表 9-3 IPv6 地址类型与地址前缀的对应关系

地址类型		地址前缀（二进制）	IPv6 前缀标识
单播地址	未指定地址	00...0 (128 bits)	::/128
	环回地址	00...1 (128 bits)	::1/128
	链路本地地址	1111111010	FE80::/10
	唯一本地地址	11111110	FC00::/7
	全球单播地址	其他形式	—
组播地址		11111111	FF00::/8
任播地址		从单播地址空间中进行分配，使用单播地址格式	

9.3.4 IPv6 地址前缀表示形式

就像 IPv4 地址可以用地址前缀表示形式一样，IPv6 地址也可以使用地址前缀表示形式，但 IPv6 地址没有“子网掩码”的概念，所以不能采用 IPv6 地址+子网掩码的表示形式。

IPv6 地址“地址前缀”表示形式是将固定不变的“网络前缀”部分保持不变，而将可变部分全部置 0，然后加上“/”符号，最后是前面“网络前缀”部分的总位数。其基本格式为：地址前缀::/前缀长度。这与 IPv4 地址中的地址前缀表示形式类似，如 IPv4 地址 192.168.1.10/24 就是地址前缀表示形式。

【注意】在以地址前缀形式表示一个 IPv6 地址时需要注意以下两点：

- 前面的“地址前缀::”部分必须包括完整的 IPv6 地址的 128 位，否则会产生异义。
- 如果要将后面可变部分的 0 用双冒号表示，则只能在 IPv6 地址网络前缀后面用一次，前

面不能再用双冒号了，因为在一个 IPv6 地址中只能用一次双冒号。

例如要描述像 12AB00000000CD3 这样一个 60 位地址前缀，则可以有以下 3 种表示形式（不同的 IPv6 地址可以表示的形式不一样，这个地址的 3 种前缀表示形式其实就是一种，只是后两种采用了压缩零规则）：

- 传统的 IPv6 地址表示形式：12AB:0000:0000:CD30:0000:0000:0000:0000/60。
- 没有采用双冒号压缩的地址前缀表示形式：12AB::CD30:0:0:0:0/60。
- 采用了双冒号压缩的地址前缀表示形式：12AB:0:0:CD30::/60。

但是以下 3 种表示形式就不行了（此时双冒号只能用在最后）：

- 12AB:0:0:CD3/60：这是因为没有补齐整个地址的 128 位，前导 0 可以不写，但后面的 0 不能省略。
- 12AB::CD30/60：仅从这种格式来展开的话，这个地址就成了 12AB:0000:0000:0000:0000:000:0000:CD30，与原来的 IPv6 地址不一样了。
- 12AB::CD3/60：仅从这种格式来展开的话，这个地址就成了 12AB:0000:0000:0000:0000:000:0000:0CD3，也不是前面的那个 IPv6 地址了。

9.4 IPv6 单播地址

IPv6 单播地址与 IPv4 单播地址一样都只标识了一个接口，用于单播通信。IPv6 单播地址有多种类型，主要包括：全球单播地址、链路本地地址、唯一本地地址（Unique Local Address，ULA）、未指定地址、环回地址。

9.4.1 IPv6 全球单播地址

“IPv6 全球单播地址”等同于 IPv4 地址中的公网地址，可以在 IPv6 Internet 上全局路由和访问。这种地址类型允许路由前缀的聚合，用来限制全球路由表项的数量。RFC3513 给出了新的 IPv6 全球单播地址的通用格式，由全球路由前缀（Global routing prefix）、子网 ID（Subnet ID）和接口标识（Interface ID）组成，如图 9-15 所示。

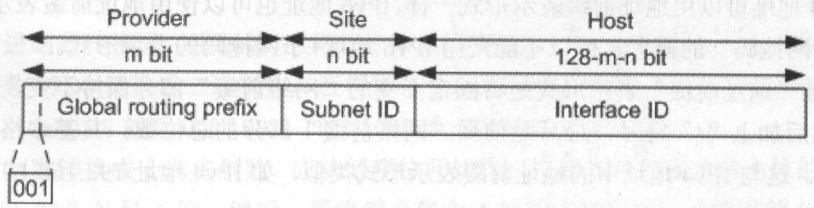


图 9-15 IPv6 全球单播地址的结构

- 全球路由前缀（Global routing prefix）：由提供商（Provider）指定给一个组织机构，通常全球路由前缀至少为 48 位。目前已经分配的全球路由前缀的最高 3 位均为 001。

- 子网 ID（Subnet ID）：子网标识，最多可分配到第 64 位（因为“地址前缀”字段最少 48 位，所以本字段最多为 16 位）。它与 IPv4 地址中的“子网 ID”作用相似，组织机构可以用子网 ID 来构建本地网络（Site）。
- 接口 ID（Interface ID）：接口标识，用来标识一个接口。

图 9-15 中也显示了全球单播地址内各组成部分组成的 3 个级别结构：“全球路由前缀”由服务提供商（Provider）分配；“子网 ID”是组织站点（Site）内的子网的集合，由用户自己分配；“接口 ID”用于标识组织站点内子网上的特定主机（Host）或接口，也由用户自己分配。

“接口 ID”是来自接口的链路层地址（MAC 地址），但这里的 MAC 地址不是我们现在 IPv4 地址中使用的 48 位 MAC 地址，而是在 IPv6 地址中扩展后的 64 位 MAC 地址——EUI-64 MAC。在一个结点的多个接口上可以使用相同的“接口 ID”，只要他们连接的是不同的子网。

对于所有 IPv6 单播地址，除了以“000”开头的那些地址外，“接口 ID”部分必须是 64 位长，而且要符合 IPv6 改进的 EUI-64 格式。64 位 EUI-64 地址是由电气和电子工程师协会（IEEE）定义的。EUI-64 地址或者被指派到网络适配器，或者从 IEEE 802 的 MAC 地址派生得到，也就是通过 EUI-64 规范生成。

IPv4 网络中的 48 位 MAC 地址的高 24 位为公司标识（高第 7 位为 0），低 24 位为扩展标识符。通过 IEEE EUI-64 规范生成“接口 ID”的方法是将 FFFE 插入 48 位 MAC 地址的公司标识和扩展标识符之间；将高第 7 位的 0 改为 1，表示此接口标识全球唯一，如图 9-16 所示。

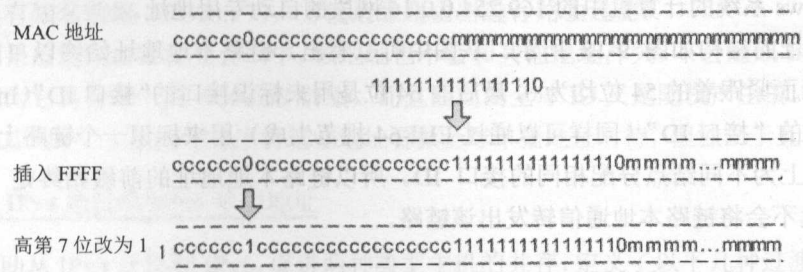


图 9-16 通过 EUI-64 规范生成“接口 ID”的流程示意图

例如 MAC 地址 000E-0C82-C4D4，转换后得到的接口标识（共 64 位）为 020E:0CFF:FE82: C4D4，如图 9-17 所示。

因特网分配地址权威机构（IANA）负责 IPv6 地址空间的分配。目前 IANA 从整个全球单播地址空间（格式前缀为 001）中取 2001::/16 进行分配。IANA 指定的注册机构把 2001:0600::/23～2001:0800::/23 段 IPv6 地址分配给欧洲和中东（包括我们中国）。这些注册机构从 IANA 得到的地址空间分配/32 前缀给 IPv6 ISP，ISP 可在/32 位前缀下继续分配/16 位前缀（相当于一共是 48 位）给每个客户，在/48 位全球路由前缀的地址空间下还可以进一步分配 16 位前缀的子网，这样每个客户最多可以有 65535 个子网。

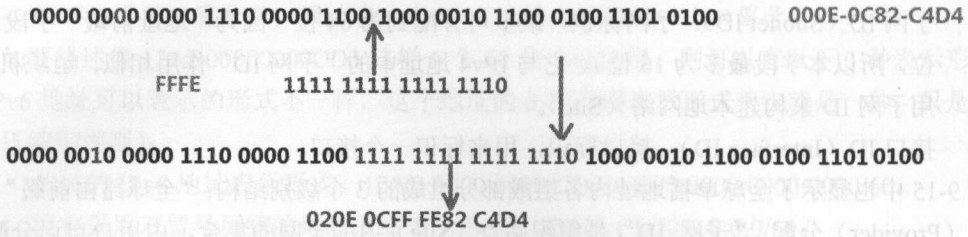


图 9-17　通过 EUI-64 规范生成“接口 ID”的示例

9.4.2　IPv6 本地单播地址

在 IPv6 地址中，本地单播地址是指本地网络使用的单播地址，也就是 IPv4 地址中经常所说的局域网专用地址。本地单播地址又有两种：“链路本地地址”和“唯一本地地址”（原来叫“站点本地地址”）。每个接口上至少要有一个链路本地单播地址，另外还可以分配任何类型（单播、任播和组播）或范围的 IPv6 地址。

1. 链路本地地址

链路本地地址只能在连接到同一本地链路的结点之间使用，固定前缀为 FE80::/10，同时将“接口 ID”添加在后面作为地址的低 64 位。启动时结点的每个接口会自动配置一个链路本地地址，但也可以采用手动配置方式。各动态路由协议邻居关系的建立所使用的就是链路本地地址，类似于在 IPv4 网络 Windows 系统的计算机中的 169.254.0.0/16 网段的自动专用地址。

链路本地地址的结构如图 9-18 所示。从图中可以看出，链路本地地址始终以 1111 1110 10（FE80）开头，后面紧跟着的 54 位均为 0，最后的 64 位是用来标识接口的“接口 ID”（Interface ID）。IPv6 单播地址中的“接口 ID”（同样可以通过 EUI-64 规范生成）用来标识一个链路上的接口，不能在同一个链路上为不同结点分配相同的接口 ID。所以链路本地地址的前缀始终是 FE80::/64。IPv6 路由器永远不会将链路本地通信转发出该链路。

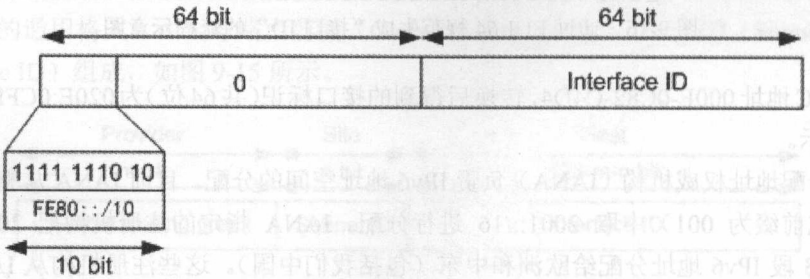


图 9-18　IPv6 链路本地地址的结构

2. 唯一本地地址

唯一本地地址（RFC4193）类似于 IPv4 地址中的私网地址，只能在本地网络内部被路由转发，而不会在全球网络中被路由转发，但上层应用程序将这些地址看做全球单播地址对待，用来替代本

地站点地址，具有知名前缀（Prefix）FC00::/7，如图 9-19 所示。

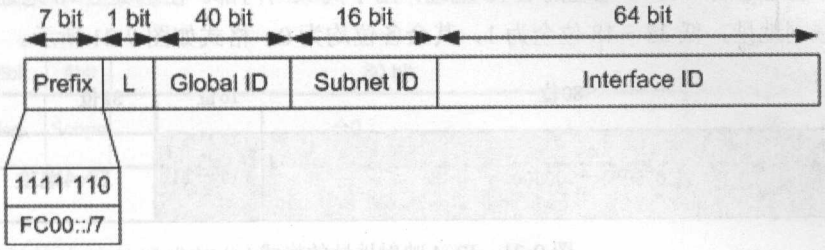


图 9-19 IPv6 唯一本地地址的结构

- Prefix（地址前缀）：固定为 FC00::/7。
- L：L 标志位，值为 1 代表该地址为在本地网络范围内使用的地址；为 0 时被保留，用于以后扩展。
- Global ID（全球 ID）：通过伪随机方式产生，参见 RFC4193。
- Interface ID（接口 ID）：接口标识，用来标识一个接口。

唯一本地地址具有以下特点：

- 具有全球唯一的前缀（虽然由随机方式产生，但是冲突概率很低）。
- 可以进行网络之间的私有连接，而不必担心地址冲突等问题。
- 具有知名前缀（FC00::/7），方便边缘路由器进行路由过滤。
- 如果出现路由泄漏到公网中，该地址也不会和其他地址冲突，不会造成 Internet 路由冲突，因为它具有唯一的 FC00::/7 前缀，而且后面的全球 ID 又是随机产生的。
- 在应用中，上层应用程序将这些地址看做全球单播地址对待，但独立于互联网服务提供商。

9.4.3 内嵌 IPv4 地址的 IPv6 单播地址

为了帮助从 IPv4 迁移到 IPv6，促进两种类型主机的并存，定义了以下几种过渡性的 IPv6 地址：

（1）IPv4 兼容地址。

在 IPv6 的转换机制中还包括了一种通过 IPv4 路由接口以隧道方式动态传递 IPv6 包的技术。这样的 IPv6 结点会被分配一个在低 32 位中带有全球 IPv4 单播地址（其余各位均为 0）的 IPv6 全球单播地址，用于 Internet 公网。这种类型的地址就称为“与 IPv4 兼容的 IPv6 全球单播地址”，其格式如图 9-20 所示。注意这里的 IPv4 必须是全球唯一的公网 IP 地址。

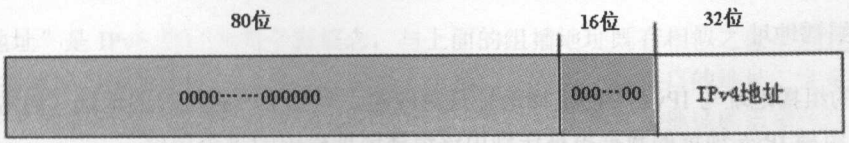


图 9-20 IPv4 兼容地址的格式

（2）IPv4 映射地址。

IPv4 映射地址是嵌入 IPv4 地址的 IPv6 地址，用于局域网内部。在这类 IPv6 地址中，低 32 位映射 IPv4 局域网地址，低 33～48 位全为 1，其余各位均为 0。格式如图 9-21 所示。

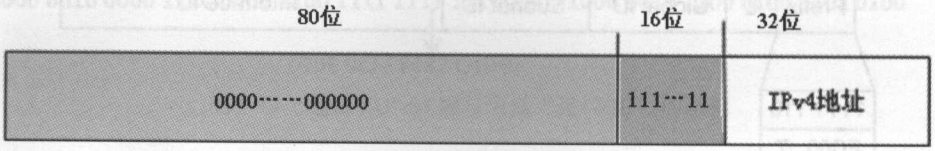


图 9-21　IPv4 映射地址的格式

（3）6to4 IPv6 地址。

6to4 IPv6 地址用于在两个通过 Internet 同时运行 IPv4 和 IPv6 的结点之间进行通信。6to4 地址是通过组合 2002::/16 前缀和该结点的公网 IPv4 地址的 32 位构成，形成一个 48 位前缀。例如，IPv4 地址 131.107.0.1，6to4 IPv6 地址前缀是 2002:836B:1::/48。当然，如果你当前使用的是动态公网 IPv4 地址，则得到的 6to4 IPv6 地址会在你每次拨号接入 Internet 时都不一样。

9.4.4　两种特殊的 IPv6 单播地址

在 IPv6 单播地址中，还有以下两种特殊的单播地址：

（1）IPv6 未指定地址。

“IPv6 未指定地址”就是 0:0:0:0:0:0:0:0/128，简写为::/128，类似于 IPv4 地址中的 0.0.0.0 地址。该地址可以表示某个接口或者结点还没有 IP 地址，可以作为某些报文的源 IP 地址，但源 IP 地址是::的报文不会被路由设备转发。

（2）IPv6 环回地址。

“IPv6 环回地址”就是 0:0:0:0:0:0:0:1/128，简写为::1/128，与 IPv4 地址中的 127.0.0.1 作用相同，主要用于设备给自己发送报文进行接口 TCP/IP 协议栈测试。实际发送的数据包中不能使用环回地址作为源 IPv6 地址或目的 IPv6 地址。

9.5　IPv6 组播地址和任播地址

对于 IPv6 地址中的组播地址和任播地址，新的 RFC3513 标准中并没有针对原来的 RFC2373 标准进行太多改动。因为这两种地址有一些相似之处，因此在一起进行介绍。

9.5.1　IPv6 组播地址

IPv6 中的组播地址与 IPv4 中的组播地址是对应的，对应于一组接口的地址（通常分属不同结点）。发送到组播 IPv6 地址的数据报被送到由该组播地址标识的每个接口。

IPv6 组播地址的最明显特征就是最高的 8 位固定为 1111 1111（有固定前缀 FF），所以 IPv6 组

播地址是很容易区分的。除了地址前缀外，组播地址还包括其他结构，以便标识它们的标志、作用域和组播组，如图 9-22 所示。

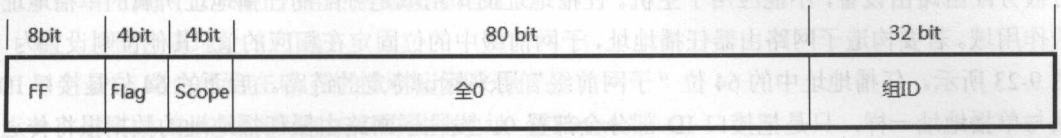


图 9-22 IPv6 组播地址的结构

组播地址结构中的各组成部分说明如下：

- Flag（标志）：占 4 位，表示在组播地址上设置的标志。目前只使用了最后一个比特（前 3 位必须置 0），当该位值为 0 时，表示当前的组播地址是由 IANA 所分配的一个永久分配地址；当该值为 1 时，表示当前的组播地址是一个临时组播地址（非永久分配地址）。
- Scope（作用域）：占 4 位，用来限制组播数据流在网络中发送的范围，各取值的具体含义如表 9-4 所示。例如，使用组播地址 FF02::2 的通信具有链路本地作用域，因为它的第 4 个十六进制位（作用域位）的值为 2。IPv6 路由器永远不会将此通信转发到本地链路以外。

表 9-4 IPv6 组播地址作用域取值的含义

取值	含义
0、3、F	保留（Reserved）
1	接口本地范围（Interface-local Scope）
2	链路本地范围（Link-local Scope）
4	管理本地范围（Admin-local Scope），即可管理的范围
5	站点本地范围（Site-local Scope）
6、7、9～D	未分配（Unassigned）
8	机构本地范围（Organization-local Scope）
E	全球范围（Global Scope）

- 组 ID：32 位，用于标识不同的组播组，并且在作用域中是唯一的。

9.5.2 IPv6 任播地址

“任播地址”是 IPv6 协议中的全新概念，与上面的组播地址既有相似之处，又有本质区别。一个 IPv6 任播地址与组播地址一样也可以识别多个接口，对应一组接口的地址。大多数情况下，这些接口属于不同的结点。但是与组播地址不同的是，目标地址是任播地址的数据包将发送给其中路由意义上最近的一个网络接口。任播地址设计的目的是用来在给多个主机或结点提供相同的服务时提供冗余功能和负载分担功能。由于使用任播地址的标准尚在不断完善中，所以目前很少有系统

支持任播传输。

目前，IPv6 中没有为任播规定单独的地址空间，任播地址和单播地址使用相同的地址空间，仅可被分配给路由设备，不能应用于主机。任播地址的作用域是分配的任播地址所属的单播地址类型的作用域。若要构造子网路由器任播地址，子网前缀中的位固定在相应的值，其他位则设置为 0，如图 9-23 所示。任播地址中的 64 位“子网前缀”用来标识特定的链路。后面的 64 位是接口 ID，语法与单播地址一样，只是把接口 ID 部分全部置 0。发往子网路由器任播地址的数据报将传送到子网中的一个路由器上。

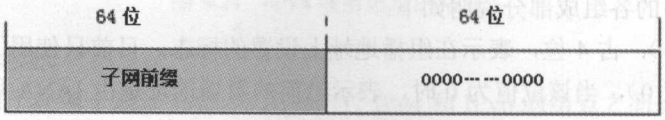


图 9-23　IPv6 任播地址的格式

9.5.3　IPv6 主机和路由器地址

由于 IPv6 地址数过于庞大，加上 IPv6 地址类型又比较复杂，所以了解我们在实际的网络配置中为主机和网络设备配置的 IPv6 地址有哪些就显得非常重要了。

IPv6 主机通常在逻辑上是多主机的，因为它们至少有两个用于接收数据报的地址：一个是用于本地链接通信的链路本地地址，一个是可路由的唯一本地地址或全球单播地址。

1. 主机可以指派的 IPv6 地址

IPv6 主机上可以指派以下单播地址：

- 用于每个接口的链路本地地址
- 用于每个接口的唯一本地地址
- 一个或多个全局单播地址
- 用于回环接口的回环地址（::1）

另外，每个 IPv6 主机都在以下组播地址侦听通信：

- 接口本地（仅指一个结点的单一接口，用于组播通信的环回测试）作用域所有结点的组播地址（FF01::1）
- 链路本地作用域所有结点的组播地址（FF02::1）
- 每个接口上用于每个单播地址的请求结点的组播地址
- 每个接口上已加入组的组播地址

2. 路由器可以指派的 IPv6 地址

在 IPv6 路由器上可以指派以下单播地址：

- 用于每个接口的链路本地地址
- 用于每个接口的唯一本地地址
- 一个或多个全局单播地址

- 用于回环接口的回环地址（::1）

IPv6 路由器可以指派有以下任播地址：

- 用于每个子网的子网路由器任播地址
- 其他任播地址（可选）

每个 IPv6 路由器将在以下组播地址侦听通信：

- 接口本地作用域所有结点的组播地址（FF01::1）
- 接口本地作用域所有路由器的组播地址（FF01::2）
- 链路本地作用域所有结点的组播地址（FF02::1）
- 链路本地作用域所有路由器的组播地址（FF02::2）
- 站点本地作用域所有路由器的组播地址（FF05::2）
- 各接口上用于每个单播地址的请求结点的组播地址
- 每个接口上已加入组的组播地址

【说明】“被请求结点的组播地址”是一种特殊的组播地址，对于结点或路由器接口上配置的每个单播地址和任播地址都会自动生成一个对应的被请求结点组播地址。

9.6 课后自测题

为了巩固本章所学的内容，请认真做下面的自测题，参考答案见附录。

9.6.1 填空题

1. IPv4 地址可分为 A、B、C、D、E 五类，其中 A 类地址的最高位固定为_____，B 类地址的最高两位固定为_____，C 类地址的最高 3 位固定为_____，D 类地址的最高 4 位固定为_____，E 类地址的最高 5 位固定为_____。

2. IPv4 广播地址类型包括 4 种，其中_____应用于一个具体的有类 IPv4 网络，_____应用于一个具体的无类 IPv4 网络子网，_____应用于一个具体 IPv4 有类网络划分的所有子网，_____应用于所有未知的有类或无类 IPv4 网络，其地址为_____。

3. 以地址前缀形式表示在 A、B、C 类 IPv4 地址中专门用于局域网的地址段分别是_____、_____、_____。

4. IPv6 地址类型包括_____、_____和_____3 种，前者又包括_____、_____、_____、_____和_____等几种类型。

5. IPv6 特殊地址包括_____和_____，其地址分别为_____和_____。内嵌 IPv4 地址的 IPv6 地址包括_____、_____和_____3 种。

9.6.2 选择题（可多选）

1. 下面关于子网掩码的说法不正确的是（ ）。

A．子网掩码可用来判断 IPv4 地址中“网络 ID”的位数

B．子网掩码中的 1 码必须从最高位开始连续

C．通过子网掩码与 IP 地址的逻辑“与”运算可以得出对应的网络地址

D．子网掩码只有 255.0.0.0、255.255.0.0、255.255.255.0 三类

2．下列关于 IPv4 地址说法不正确的是（　　）

A．C 类 IPv4 地址就是局域网用的 IPv4 地址

B．A 类 IPv4 网络共 128 个

C．IPv4 地址中“网络 ID”部分不能以数字 0 或 127 开头

D．不能给主机分配“主机 ID”部分全 0 或全 1 的 IPv4 地址

3．下面（　　）不是无类 IPv4 地址所带来的好处。

A．提高 IPv4 地址的利用率　　B．提高路由精细度

C．增加可用 IPv4 地址数　　D．提高网络的安全性

4．一个 B 类单播 IPv4 网络中，可以分配给主机的 IP 地址数是（　　）。

A．65536　　B．65534　　C．254　　D．16777216

5．以下有类 IPv4 地址中属于 B 类地址的是（　　）。

A．112.213.12.23　　B．210.123.23.12　　C．23.123.213.23　　D．156.123.32.12

6．11100110 10011101 00110011 01110001 对应的十进制 IPv4 地址是（　　）。

A．230.157.51.113　　B．230.157.50.132

C．232.157.51.113　　D．232.156.50.112

7．以下有类 IPv4 地址中属于 A 类地址的是（　　）。

A．10110100 00110011 01100010 10101100　　B．10001110 11011011 11100000 10001100

C．01001010 01100011 01100000 10011000　　D．11011110 11100011 01100100 00001100

8．以下不能用来标识主机接口 IPv4 地址的是（　　）。

A．10110100 00110011 01100010 10101100　　B．11101110 11011011 11100000 10001100

C．01001010 01100011 01100000 10011000　　D．11111110 11100011 01100100 00001100

9．下列关于网络地址和广播地址说法正确的是（　　）。

A．网络地址的“网络 ID”部分一定全为 0

B．广播地址的“主机 ID”部分一定全为 1

C．网络地址的最后一个字节值一定为 0

D．广播地址的最后一个字节值一定为 255

10．下列关于 IPv4 地址前缀说法正确的是（　　）。

A．地址前缀值代表了对应 IPv4 地址中“网络 ID”的位长度

B．地址前缀值代表了对应 IPv4 地址中“主机 ID”的位长度

C．地址前缀值代表了对应 IPv4 地址所对应的子网掩码中连续 1 的位长度

D．地址前缀值越大，则对应 IPv4 网络所拥有的 IP 地址数越多

11. 在没有划分子网的情况下，IP 地址 125.3.54.56 所在网络的网络地址是（ ）。

A. 125.0.0.0 B. 125.3.0.0 C. 125.3.54.0 D. 125.3.54.32

12. 下列有类 IPv4 地址中可分配给主机使用的是（ ）。

A. 127.2.3.5 B. 1.255.255.2

C. 255.23.200.9 D. 192.240.150.255

13. 下列无类 IPv4 网络中可分配给主机使用的是（ ）。

A. 10.0.16.0/24 B. 10.0.16.0/16

C. 192.168.0.255/23 D. 192.168.31.255/21

14. 以下不属于局域网专用 IPv4 地址的是（ ）。

A. 10.0.0.1 B. 172.31.254.1 C. 192.168.253.16

D. 172.14.230.10 E. 12.120.12.1

15. 下列组播 IPv4 地址中用于分配给本地组播网络中所有支持组播通信的主机的是（ ）。

A. 224.0.0.0 B. 224.0.0.1 C. 224.0.0.2

D. 224.0.0.4 E. 224.0.0.5

16. 下列组播 IPv4 地址中用于分配给本地组播网络中所有 RIP 路由器的是（ ）。

A. 224.0.0.6 B. 224.0.0.9 C. 224.0.0.10

D. 224.0.0.13 E. 224.0.0.22

17. 在 192.168.2.16/28 子网中每个子网最多可以容纳（ ）台主机。

A. 16 B. 8 C. 15 D. 14

18. 某公司申请到一个 C 类 IPv4 公网地址，但要连接 9 个子公司（每个子公司单独一个子网），则子网划分后的最合适的子网掩码是（ ）。

A. 255.254.255.240 B. 255.255.255.192

C. 255.255.255.128 D. 255.255.255.240

19. 与子网掩码为 255.255.255.224 的 IPv4 地址 10.110.12.29 在同一子网的主机 IPv4 地址是（ ）。

A. 10.110.12.0 B. 10.110.12.32

C. 10.110.12.31 D. 10.110.12.30

20. 子网掩码为 255.224.0.0 的子网 5.96.0.0 所允许的最大主机 IPv4 地址是（ ）。

A. 5.127.255.254 B. 5.128.255.254

C. 5.96.255.254 D. 5.159.255.254

21. 如果子网掩码为 255.255.248.0，则其对应的网络地址可以是（ ）。

A. 150.150.42.0 B. 150.150.15.0

C. 150.150.16.0 D. 150.150.96.0

22. 一个 C 类网络划分子网后的掩码为 255.255.255.192，则每个子网包含的子网 ID 位数、子网数目、主机地址数依次是（ ）。

A．2，4，62　　B．3，8，30　　C．3，14，30　　D．5，8，6

23．一IPv4地址为172.168.120.1/20，则该地址所在子网的网络地址和子网掩码是（　　）。

A．172.168.112.0，255.255.240.0　　B．172.168.96.0，255.255.224.0

C．172.168.120.0，255.225.240.0　　D．172.168.120.0，255.255.224.0

24．一子网的网络地址为126.68.24.0，子网掩码为255.192.0.0，则该子网的广播地址是（　　）。

A．126.68.24.255　　B．126.64.255.255

C．126.127.255.255　　D．126.255.255.255

25．要实现对 192.168.4.0/24～192.168.7.0/24 四个子网进行聚合，则聚合路由的子网掩码为（　　）。

A．255.255.255.0　　B．255.255.0.0

C．255.255.254.0　　D．255.255.252.0

26．在内部源地址的NAT转换中，内部网络发送的数据包的目的IP地址是（　　）。

A．内部本地地址　　B．外部本地地址

C．内部全局地址　　D．外部全局地址

27．在内部源地址的NAT转换中，外部网络发送的回程数据包的源IP地址是（　　）。

A．内部本地地址　　B．外部本地地址

C．内部全局地址　　D．外部全局地址

28．下面关于IPv6地址说法正确的是（　　）。

A．IPv6地址常用的表示形式也是十进制

B．IPv6地址中包括8段，16个字节

C．IPv6地址各个段之间用小圆点（.）分隔，共有128位二进制

D．IPv6地址主要分为单播、多播和广播3种类型

29．下面关于IPv6地址1000:E900:0000:0000:02AA:0000:FE28:9C5A的表示正确的是（　　）。

A．1000:E900:0:0:02AA:0:FE28:9C5A　　B．1000:E900::02AA:0:FE28:9C5A

C．1000:E900::02AA::FE28:9C5A　　D．1:E9::02AA:0:FE28:9C5A

30．如果想要表示 E10200000000683096F0 这样一个 72 位地址前缀，则以下格式正确的是（　　）。

A．E102:0:0:6830:96F0:0:0:0/72　　B．E102::6830:96F0::/72

C．E102:: 6830:96F0/72　　D．E102:0:0:6830:96F0::/72

E．E102:0:0: 6830:96F::/72

31．下面关于全局单播IPv6地址说法不正确的是（　　）。

A．最高4位固定为0010

B．最低的64位一般是对应接口的EUI-64地址

C．是可以在互联网直接路由的公网IPv6地址

D．目前我国可以使用的全局单播IPv6地址段为2001:0600::/23～2001:0800::/23

32. 链路本地单播 IPv6 地址的地址前缀形式为（ ）。

A. FFC0::/10 B. FEC0::/48 C. FE80::/10 D. FE80::/64

33. 唯一本地单播 IPv6 地址的地址前缀形式为（ ）。

A. FFC0::/10 B. FC00::/7 C. FE80::/10 D. FE80::/64

34. 下面的 IPv6 地址中与 IPv4 局域网专用地址对应的是（ ）。

A. 全局单播 IPv6 地址 B. 链路本地单播 IPv6 地址

C. 唯一本地单播 IPv6 地址 D. IPv4 映射地址

35. 以下 IPv6 地址用于互联网中的是（ ）。

A. 全局单播 IPv6 地址 B. 6to4 IPv6 地址

C. IPv4 兼容地址 D. IPv4 映射地址

36. 以下 IPv6 地址的最高 96 位全为 0 的是（ ）。

A. 全局单播 IPv6 地址 B. 唯一本地单播 IPv6 地址

C. IPv4 兼容地址 D. IPv4 映射地址

37. 以下 IPv6 地址的最高 80 位全为 0 的是（ ）。

A. 全局单播 IPv6 地址 B. 唯一本地单播 IPv6 地址

C. IPv4 兼容地址 D. IPv4 映射地址

38. 如果一个公网 IPv4 地址为 202.10.196.2，现在转换成 6to4 类型的 IPv6 地址，则其前缀为（ ）。

A. 2002:CE0A:8402::/48 B. CE06:8402::/32

C. 2002:CE06:8402::/48 D. CE0A:8402::/32

39. 下面关于组播 IPv6 地址的说法不正确的是（ ）。

A. 组播 IPv6 地址的最高 8 位固定为 11111111

B. 组播 IPv6 地址可识别多个接口，对应于一组接口的地址

C. 组播地址可以把数据发送到该地址标识范围中的多个目的点

D. 组播 IPv6 地址中的链路本地作用域值为 0001

40. 下面的组播 IPv6 地址中属于链路本地作用域所有结点的是（ ）。

A. FF02::1 B. FF02::2 C. FF01::2 D. FF01::1

41. 下面关于 IPv6 任播地址说法不正确的是（ ）。

A. 任播地址只能分配给子网路由器，而不能分配给主机使用

B. 通过任播地址发送数据时，可以把数据发送到多个接口上

C. 任播地址与组播地址一样可以标识多个接口

D. 任播地址中除“子网前缀”外的其他部分都置 0

42. 下面可以在主机上配置的 IPv6 地址是（ ）。

A. 链路本地地址 B. 唯一本地地址

C. FF01::1 D. 任播地址

43．管理员将要在路由器 G0/0/1 接口上配置 IP 地址，那么使用下面的（　　）才有可能得到正确的配置。

A．192.168.10.112/30　　B．237.6.1.10/24

C．145.4.2.55/26　　D．127.3.1.4/28

44．在一个子网掩码为 255.255.240.0 的网络中，（　　）是合法的网络地址。

A．150.150.0.0　　B．150.150.0.8　　C．150.150.8.0　　D．150.150.16.0

45．已知某个子网的子网掩码是 255.255.248.0，那么下列地址中同属于该网段的是（　　）。

A．10.110.16.1 和 10.110.25.1　　B．10.76.129.21 和 10.76.137.1

C．10.52.57.34 和 10.52.62.2　　D．10.33.23.2 和 10.33.31.1

46．某公司使用一个 C 类 IP 地址段，但要分配给 6 个子公司，最大的一个子公司有 26 台计算机，不同的子公司必须在不同的网段，则子网掩码应该设置为（　　）。

A．255.255.255.0　　B．255.255.255.128

C．255.255.255.192　　D．255.255.255.224

47．一个网段 150.25.0.0 的子网掩码是 255.255.224.0，那么下列地址中是该网段中有效主机地址的是（　　）。

A．150.25.0.0　　B．150.25.1.255　　C．150.25.2.24　　D．150.15.3.30

48．在一个 C 类地址的网段中要划分出 32 个子网，最合适的子网掩码是（　　）。

A．255.255.255.254　　B．255.255.255.240

C．255.255.255.255　　D．255.255.255.248

49．路由条目 10.0.0.24/29 可能由以下（　　）子网路由聚合而来。

A．10.0.0.25/30　　B．10.0.0.23/30　　C．10.0.0.26/30　　D．10.0.0.22/30

50．IP 地址 199.32.59.64 的子网掩码为 255.255.255.224，则其网络地址为（　　）。

A．199.32.59.64　　B．199.32.59.65　　C．199.32.59.192　　D．199.32.59.224

51．IP 地址 192.168.1.7/28 的子网掩码为（　　）。

A．255.255.255.240　　B．255.255.255.248

C．255.255.255.224　　D．255.255.255.252

52．一个子网的网络地址为 5.32.0.0，子网掩码为 255.224.0.0，则最大主机地址是（　　）。

A．5.32.254.254　　B．5.32.255.254　　C．5.63.255.254　　D．5.63.255.255

53．一个由 C 类网络划分的子网的子网掩码为 255.255.255.224，则包含的子网 ID 位数、子网数、每个子网中的主机数为（　　）。

A．2，4，62　　B．3，8，30　　C．4，16，14　　D．5，32，6

54．10.1.0.1/17 的广播地址是（　　）。

A．10.1.128.255　　B．10.1.63.255　　C．10.1.127.255　　D．10.1.126.255

55．网段：172.128.12.0,172.128.17.0,172.128.18.0,172.128.19.0，最好用下列（　　）网段实现路由汇总。

A．172.128.0.0/21　　B．172.128.0.0/19

C．172.128.12.0/22　　D．172.128.20.0/20

56．下列可使用 CIDR 技术正确完成路由聚合的是（　　）。

A．192.168.1.0～192.168.15.0 聚合成 192.168.0.0/19

B．192.168.1.0～192.168.15.0 聚合成 192.168.0.0/20

C．192.168.1.0～192.168.15.0 聚合成 192.168.0.0/21

D．192.168.1.0～192.168.15.0 聚合成 192.168.0.0/22

57．管理员计划将 172.16.4.32/29 网段中的 IP 地址分配给主机，下列选项中的（　　）IP 地址是管理员可以分配给主机的 IP 地址。

A．172.16.4.32　255.255.255.240　　B．172.16.4.33　255.255.255.248

C．172.16.4.34　255.255.255.240　　D．172.16.4.35　255.255.255.248

58．某路由器上 G0/0/1 接口的 MAC 地址是 00E0-FC03-AA73，此接口的 IP 地址是 2001::2E0:FCFF:FE03:AA73，由此可以判断出此接口的接口标识是基于（　　）得到的。

A．DHCPv6　　B．Auto-link　　C．ARP　　D．EUI-64

59．下面（　　）是 IPv6 地址 2000:002A:0000:0000:0000:000A:020C:023D 的正确压缩格式。

A．2000:2A::A:20C:23D　　B．2000:2A:0:0:0:A:20C:23D

C．2000:2A::A:2C:23D　　D．2000:2A:0:0:0:A:2C:23D

60．IPv6 主机 A 要与 IPv6 主机 B 通信，但不知道主机 B 的链路层地址，遂发送邻居请求消息，邻居请求消息的目的地址是（　　）。

A．广播地址　　B．全部主机组播地址

C．主机 A 的被请求结点组播地址　　D．主机 B 的被请求结点组播地址

10 传输层

“传输层”与“数据链路层”的作用都是建立数据传输通道，两者在功能上存在许多相似之处。但是传输层的功能主要体现在广域网网络应用中，可以把“数据链路层”当作局域网通信的数据传输通道，而“传输层”则是广域网中的数据传输通道。本章也主要针对广域网中的传输层来进行各方面功能原理的介绍。但是，在局域网中的网络应用同样需要用到传输层功能，也是由用户计算机操作系统中的 TCP 或 UDP 协议来完成的，不同的只是在局域网中传输连接是永久连接的，因为在局域网中的数据链路是永久连接的，又不需要与其他网络进行连接。

“传输层”是整个广域网体系结构模型中的核心所在，因为它负责的端到端的通信是面向网络通信的低三层和面向信息处理的最高三层之间的中间一层，起到桥梁作用。“传输层”同时是两台计算机系统经过网络进行数据通信时第一个端到端的层次。本章主要介绍 OSI/RM 中传输层协议和 TCP/IP 协议体系结构中 TCP 协议的各方面功能（主要包括传输连接的建立/释放、数据传输、差错控制、流量控制、拥塞控制等）及实现原理。

本章重点与难点：

- OSI/RM 体系结构传输连接建立、数据传输和传输连接释放各阶段的主要 TPDU
- OSI/RM 体系结构传输层主要的服务原语
- OSI/RM 体系结构传输层寻址方案及原理
- OSI/RM 体系结构传输连接建立、数据传输和传输连接释放流程
- OSI/RM 体系结构流量控制方案及原理
- TCP 协议数据段格式
- TCP 协议的可靠性传输机制及原理
- TCP 协议的流量控制方案及原理

- TCP 协议的拥塞控制方案及原理
- UDP 协议数据报格式

10.1 传输层基础

在整个计算机网络体系结构中，通常是将 OSI/RM 七层模型中的下面三层称为面向通信子网的层，负责通信通道的建立，而将传输层及以上的各层称为面向资源子网的层，负责终端系统间的数据通信。还有一种划分是将传输层及以下的三层统称为面向通信的层，总体来说是负责通信通道的建立和数据传输，而将传输层上面的会话层、表示层和应用层统称为面向应用的层，如图 10-1 所示。

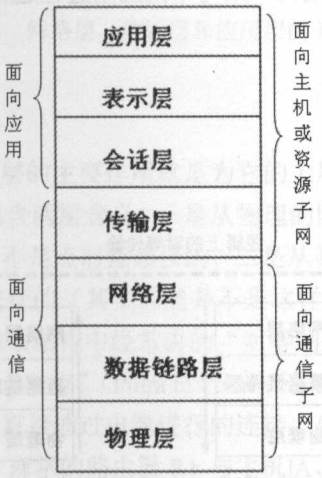


图 10-1 传输层在 OSI/RM 模型中的特殊位置

10.1.1 划分传输层的必要性

从通信和信息处理两方面来看的话，“传输层”既是面向通信部分的最高层，它与下面的三层一起共同构建进行网络通信所需的线路和建立数据传输通道，同时又是面向用户的最低层，因为无论何种网络应用，最终都需要把各种数据报传送到对方。来自应用层的用户数据必须依靠传输层协议在不同网络中的主机间进行传输，因为仅靠网络层把数据传送到目的主机上还是不够的，还必须把它交给目的主机的应用进程。所以总的来说，无论是在哪种划分方式中，“传输层”的位置都非常特殊，都是起到一个承上启下的桥梁作用。

接着我们再从数据通信原理上具体分析各层的基本作用来看划分“传输层”的必要性。我们知道，物理层为数据通信提供实际的物理线路和通信信道，这是所有数据通信的基础；数据链路层为同一网络中（数据链路层的通信限于同一局域网中）的数据通信提供了虚拟的通信通道，可以根据

不同的链路类型对物理层的比特流进行帧封装和传输；网络层为不同网络间的数据通信提供了数据包的路由、转发功能，把数据包从一个网络中的主机选择最佳路径传送到位于另一网络中的目的主机上。这时可能就有读者会问到，既然网络层已把源主机上发出的数据包传送给了目的主机，那么为什么还需要设置一个传输层呢？这需要我们理解主机间用户应用层通信的实体。

位于两网络中主机间的真正数据通信主体不是这两台主机，而是两主机中的各种网络应用进程。因为在同一时刻，两主机间可以进行多个应用通信。例如某两个用户在进行视频通信时，还进行了 QQ 聊天，或者还要同时打网络电话，这就是多个应用进程，如图 10-2 所示（仅列出了一个方向）。而不同通信的标识就是进程 ID，只有通信进程才可以把对应的数据包传输到对应的应用中。正因为如此，在进行具体的网络通信应用时必须为每一个网络应用配备一个唯一的应用进程标识，否则所传输的报文就不知道要提交给哪个用户应用了。而这里的应用进程识别就是依靠本章所要介绍的“传输层”了，它就是通过“端口”来与不同应用进程进行对应的。当然这是后话，具体将在本章后面介绍。

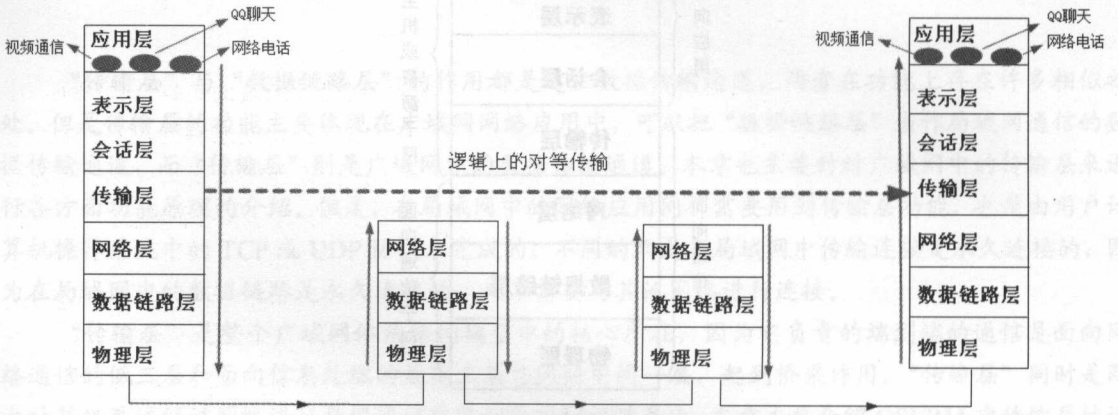


图 10-2　多应用进程的网络通信流程示意图

不同网络间主机的整体通信过程基本如下：从一个用户主机的应用层发出的应用请求报文到了传输层后在数据的头部添加对应的传输层协议头部信息（具体将在本章后面介绍），封装成数据段，然后传到网络层后被封装在报文（或者报文分组）中的“数据”部分，然后再依次传到数据链路层，重新封装成数据帧，最后通过物理层以比特流的方式一位位地向对方网络传输。到了对方网络中，数据沿着与发送端相反的方向进行解封装，然后依次传输到对应的传输层，最后根据各通信进程中的标识提交给应用层中的相应应用进程。如果中间经过多个路由器，则这些中间路由器只进行最低三层的报文封装和解封装以及其他对应功能，不建立基于传输层的对等连接关系。

“传输层”是源端到目的端从低到高对数据传送进行控制的最后一层，最终目标就是向它的用户（应用层中的应用进程）提供高效、可靠和性价比合理的服务。在传输层内部，完成这项任务的硬件或软件称为“传输实体”（Transport Entity）。传输实体可能位于操作系统的内核，或者在一个独立的用户进程中，或者以一个链接库的形式被绑定在具体的网络应用中。网络层、传输层和应用

层之间的关系如图 10-3 所示（其中的 TPDU 为传输协议数据单元）。但是只有资源子网中的终端设备才具有传输层功能，通信子网中的设备一般至多只具备 OSI/RM 下面三层的功能，即通信功能。

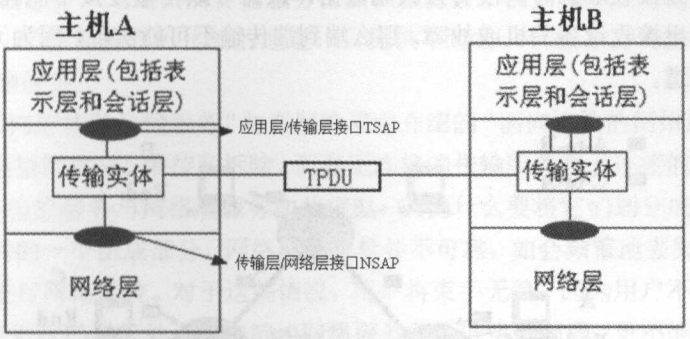


图 10-3 网络层、传输层和应用层的关系

10.1.2 传输层的端到端传输服务

我们经常听到别的同行说，传输层的主要作用就是为它的上层提供端到端的数据传输服务。那么什么叫端到端服务呢？其实这里包含两层含义：一是从物理的网络连接角度来讲的，端到端是指网络通信的双方不是在同一链路上，不是点对点连接的；二是从数据传输连接角度来讲的，端到端是指在用户看来两端的连接是直接进行的（其实当然并不是这样的），屏蔽了核心网络结构和各种子网间的差异。当然，其实这也是体系结构中经常讲的“对等通信”原理。

在这里就涉及到两个概念，即“点对点”（Point to Point）连接和“端到端”（End to End）连接。所谓“点对点”连接就是通信双方是直接通过电缆进行的连接，也就是通常所说的背对背连接，中间没有经过任何其他设备。如图 10-4 所示的路由器 R1 与主机 A、路由器 R2 与主机 B、路由器 R3 与主机 C 之间的连接，以及路由器 R1 与路由器 R2、路由器 R1 与路由器 R3、路由器 R2 与路由器 R3 之间的连接都属于点对点连接，因为在这些连接链路的中间没有任何其他设备。而“端到端”连接是两个终端系统之间的连接，体现在两个终端系统的连接中要经过一个或多个设备。如图 10-4 所示的主机 A 与主机 B、主机 C 与主机 A、主机 B 与主机 C 之间的连接都属于“端到端”连接，它们之间都经过了一个或多个路由器的转连。

与“点对点”连接和“端到端”连接相对应的就是“点对点”传输和“端到端”传输。“端到端”传输在数据传输前需要在两端所经过的线路上建立一条构建于网络层已搭建好的路由路径之上的虚拟传输连接（当然这里仅是针对面向连接的传输层协议而讲的），以此来屏蔽所经网络类型或参数配置的不同，就像它们是直接相连的一样；链路建立后，源端即可发送数据，直至数据发送完毕，接收端确认接收成功。

“端到端”传输的优点是链路建立后源端知道接收设备一定能收到（这就是“可靠性”的体现），而且经过中间交换设备时无需建立多条传输链路，节省了设备资源。“端到端”传输的缺点是直到目的端收到数据为止源端的设备一直要参与传输，因为只有在传输完成后从源端到目的端所建立的

虚拟传输连接才能拆除，否则会中途中断数据的传输。这样一来，如果整个传输的延迟很长，那么会对源端的设备造成很大浪费，因为那时源端可能早就把所有数据都发完了，只是因为中间子网的一些延迟原因而导致接收端长时间没有接收完数据，最终导致传输连接不能释放。“端到端”传输的另一个缺点是如果接收设备关机或故障，那么端到端传输不可能实现，因为无法在两端建立一条虚拟的专用传输通道。

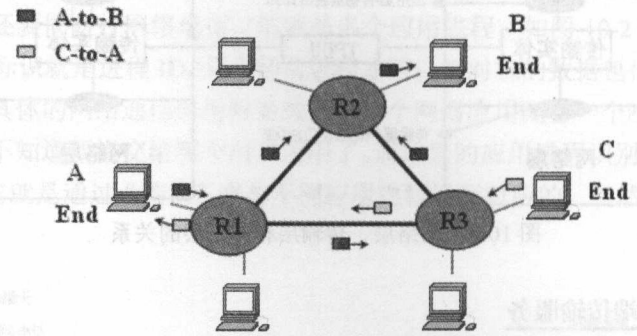


图 10-4 “点对点”连接与“端到端”连接示例

“点对点”传输使源端可以直接把数据传给与它直接相连的设备，在需要的时候又可以把数据传给与之直接相连的下一台设备，通过一台台直接相连的设备把数据传到接收端。很显然这是一种接力传输方式。“点对点”传输的优点是源端设备送出数据后，它的任务已经完成，不需要参与整个传输过程，这样不会浪费源端设备的资源，后面的数据传输是由后面的设备来完成的。另外，即使接收端设备关机或故障，点对点传输也可以采用存储转发技术进行缓冲，当然缓冲也是有大小和时间限制的。“点对点”传输的缺点是源端发出数据后，不知道目的端能否收到或何时能收到数据，因为它不需要事先与对方建立起连接，数据发送也是以数据报方式进行的，不管对方是否工作正常都把数据发送过去。

在一个网络系统的不同层次中，可能用到端到端传输，也可能用到点对点传输。如 Internet 网，网络层及以下各层都采用点对点传输，传输层以上采用端到端传输。

10.1.3 传输层服务

“传输层”在通信子网提供的服务基础上，为源主机和目的主机之间提供可靠的、透明的、可由用户选择的数据传输服务，是面向资源子网的高层和面向通信子网的低层之间的软件接口或者桥梁。同时，传输层也弥补了高层所要求的可靠、可控制服务与网络层所提供的非可靠服务之间的差距，并向高层屏蔽通信子网中不同网段的差异。

1. 传输层服务类型

与网络层有面向连接服务和无连接服务两大类一样（在 TCP/IP 体系结构中只有无连接的网络层服务，因为只有 IP 这一种网络层连接协议），传输层的服务也有这两大类。面向连接的传输服务

在提供传输服务前需要先建立专门的传输连接，而且这条连接是可管理的，在需要时或者通信结束时进行拆除。面向连接的传输服务是可靠的传输服务，而且可提供拥塞控制和差错控制功能，如 TCP 协议提供的传输服务。无连接的传输服务在提供服务前不需要建立专门的传输连接，直接向目的节点发送数据，不管是否有可传输的通道，只提供不可靠（仅做尽力传输）的传输服务，如 UDP 协议提供的传输服务。

此处所说的“面向连接的传输服务”与在网络层中介绍的“面向连接的网络层服务”十分相似，两者都向用户提供连接的建立、维持和拆除，而且无连接的传输服务与无连接的网络层服务也十分相似。那么，既然传输层服务与网络层服务如此相似，又为什么要将它们划分成两个层次呢？原因是网络层是通信子网的一个组成部分，网络服务质量并不可靠，如会频繁地丢失分组、网络层系统可能崩溃或不断地进行网络复位。对于这些情况，用户将束手无策，因为用户不能对通信子网加以控制，因而无法采用更优的通信处理机来解决网络服务质量低劣的问题，更不能通过改进数据链路层纠错能力来改善它。解决这一问题的唯一可能办法就是在资源子网的主机上增加一个可以由用户控制的传输层。通过传输层的面向连接服务使得数据传输更加可靠，特别是对于 IP 这类无连接网络层服务。

2. 服务等级

不同类型的网络以及不同连接性能的网络，所提供的传输层服务级别也不一样。

传输层提供的服务可分为“传输连接”服务和“数据传输”服务。“传输连接”服务是传输层为它的上层发出的每个应用进程在网络层提供的服务基础之上建立一个相应的逻辑连接；而“数据传输”服务则是在所建立的传输连接基础上为用户提供比网络层更有保障（主要针对面向你的传输层服务而言）、更有针对性并且能由用户自己进行拥塞控制、差错控制的数据传输。

传输层所能提供的服务质量水平不仅与所选择的传输层协议有关，更重要的是与所服务的网络类型有关，毕竟传输层是建立在网络层基础之上的，依赖网络层所提供的路由服务。根据不同网络所提供的不同传输层服务质量水平，可以把传输层服务分为以下三大类：

- A 类：具有低的连接差错率和故障率，是可靠的传输服务，一般指虚电路网络上。
- B 类：具有较低的连接差错率和故障率，传输服务质量中等，应用于广域网和互联网中的传输层协议多数属于这一类。
- C 类：具有较高的连接差错率和故障率，服务质量较差，数据报服务或无线电分组交换网所提供的传输层协议均属此类。

3. 传输层协议类别

在OSI/RM体系结构的传输层中，不同类型的传输层协议所支持的服务级别不一样，共分TP0～TP4 五种类别。其中 TP0～TP3 这 4 种类别均属于面向连接的传输层协议，发送数据前必须先建立传输连接；TP4 既可以是面向连接的传输层协议，也可以是无连接的传输层协议。

- TP0：简单类，只建立一个简单的端到端传输连接，提供基本的数据段和重组功能。TP0 传输层协议先识别网络层所支持的最大传输单元（MTU）大小，然后据此必要时对数据包进行数据分段，在接收端再对数据段进行重组。

- TP1：基本差错恢复类，在 TP0 类的基础上新增了差错控制功能，可以对每个 TPDU 进行编号，当收到不确认（NAK）消息或者有 TPDU 丢失时将会重发；如果有太多的不确认 TPDU，TP1 类别传输层协议还可重新初始化传输连接。
- TP2：多路复用类，在 TP0 的基础上新增了传输连接的多路复用和解复用功能，以及流量控制功能。
- TP3：差错恢复和多路复用类，是 TP1 和 TP2 传输层协议功能的综合。
- TP4：差错检测和恢复类，在 TP3 的基础上新增了差错检测功能，是最复杂的传输层协议类别。在功能上类似于 TCP/IP 体系结构中的 TCP 协议，但 TP4 类别传输层协议既支持面向连接的网络服务，又支持无连接的网络服务。

以上这 5 种传输层协议类别的综合比较如表 10-1 所示。

表 10-1　5 种传输层协议类别的综合比较

协议类别	主要功能	传输层协议类型
TP0	数据分段和重组	面向连接的传输层协议
TP1	数据分段和重组、差错控制	
TP2	数据分段和重组、多路复用和解复用	
TP3	数据分段和重组、差错控制、多路复用和解复用	
TP4	数据分段和重组、差错控制、多路复用和解复用、差错检测	既可以是面向连接的传输层协议，又可以是无连接的传输层协议

10.1.4　TSAP 和 TPDU

在 OSI/RM 传输层中涉及到两个非常重要的术语，即 TSAP（传输服务访问点）和 TPDU（传输协议数据单元），它们贯穿了整个传输层数据传输服务提供的始末，下面进行具体介绍。

1. TSAP

在计算机网络中，传输层要在用户之间提供可靠、有效的端到端服务，必须把一个用户进程和其他的用户进程区分开，这个工作主要是由传输层地址来实现。我们在第 6 章介绍数据链路层时就已经说到，不同层之间的互访是通过 SAP（服务访问点）进行的。SAP 位于除最高的应用层外其他各层的上边缘，用作为下层向上层提供服务、上层调用下层服务的逻辑接口。在传输层的上边缘也有对应的 SAP（而且可能远不止一个），那就是 TSAP（传输服务访问点）。TSAP 就是上层调用传输层服务、传输层为它的上层提供服务的逻辑接口，是传输层地址，参见图 10-2。

同一时间同一对网络实体间的用户应用进程可能有多个，所以不能仅靠网络实体地址（即 NSAP）来标注通信双方（因为此时通信的实体是各个应用进程，而不是通信双方主机），而必须借助传输层地址进行标识。TSAP 就相当于传输层的地址，不同的 TSAP 标识不同的会话或应用进程。

为确保所有的传输地址在整个网络中是唯一的，传输地址规定由网络 ID、主机 ID 和主机分配

的传输层端口三部分共同组成。前面“网络 ID”和“主机 ID”这两部分正好是 IP 地址的两部分，用来指定具体的网络实体，是 NSAP（网络服务访问点）；后面的“传输层端口”是传输层特定的属性，用来与应用进程进行一一对应，所以说真正的传输层地址其实就是具体应用所占用的传输层端口。

在这里不得不说的是，在总共 65535 个对应传输层协议端口中，前 1024 个端口称为常规端口，代表了特定的应用层服务，或者说是应用层进程的，不能分配给其他应用服务使用（其他的端口可以随意使用），而且同一时间同一主机上不能有两个应用进程使用相同的端口号，所以在为应用进程分配端口时也不能随意。如 80 号 TCP 端口是分配给 WWW（对应 http 应用）服务使用的，23 号 TCP 端口是分配给 Telnet 服务的，21 号 TCP 端口是分配给出 FTP 服务的，……。

2. TPDU

在第 5 章中也已经介绍到，在分层的网络体系结构中每一层都是使用 PDU（Protocol Data Unit，协议数据单元）来与对方的对等层进行通信的，PDU 包含来自上层的信息以及当前层实体附加的信息。然后这个 PDU 被传送到下一较低的层。其实在其他层中我们都提到了 PDU 这个术语，只不过到了最后我们不是直接说是 xPDU，而是为各层起了一个专用的名称，如数据链路层中 LLC 和 MAC 两个子层中所说的“帧”就分别对应了 LPDU（逻辑链路协议数据单元）和 MPDU（介质访问协议数据单元），网络层中所说的“分组”或“包”其实对应的就是 NPDU（网络协议数据单元）。但到了传输层，它的数据单元是“数据段”，这个名称显然太容易混淆，于是在 OSI/RM 传输层中我们还是把其中传输的数据单元称为 TPDU（不过在 TCP/IP 体系结构中，TCP 协议的协议数据单元仍然为“数据段”）。所以不要一下子见到这里来一个 TPDU 就不适应了，其实与本书前面几章中介绍的“比特”“帧”“分组”（或“包”）是同类概念。

既然 TPDU 是传输层的协议数据单元，那么它与数据链路层的“帧”、网络层的“分组”一样，一定也有它的基本格式，那就是由“TPDU 报头”和“TDPU 有效载荷”两部分组成，与其他层中的 PDU 基本格式一样。其中“TDPU 有效载荷”就是来自会话层（OSI/RM 体系结构中）或者应用层（TCP/IP 体系结构中）的数据，而“TPDU 报头”又包含“长度指示”（LI）“固定部分”和“可变部分”这 3 个基本组成部分。整个 TPDU 的基本结构如图 10-5 所示。

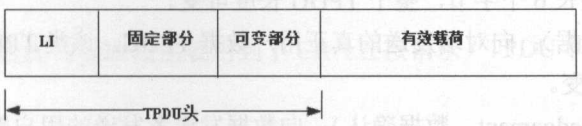


图 10-5 TPDU 基本结构

- LI（长度指示）：占 1 个字节，以字节为单位标识整个 TPDU 头部的长度，最大值为 254。
- 固定部分：包括 TPDU 代码和常用参数，其内容对于某一特定的 TPDU 是固定的，如包括目的端口和源端口等，但对于不同传输层协议的 TPDU 是不一样的。
- 可变部分：包含一些可选参数项，每个参数项由 3 个字段：Code（代码，1 字节）、Length（长度，1 字节）和 Value 值（参数值，可变长度）构成，也就是通常所说的 CLV。因为

各个 TPDU 头部的可变部分所包含的参数项数目不同，所以其长度是可变的。OSI/RM 中定义的常见 TPDU 类型参数以及它们对应的代码值如表 10-2 所示。

表 10-2　OSI/RM 中定义的常见 TPDU 类型

TPDU 类型	名称	适用的传输协议类别					代码	可携带的有效载荷大小
		0	1	2	3	4		
CR	连接请求	√	√	√	√	√	1110	≤32 字节
CC	连接确认	√	√	√	√	√	1101	≤32 字节
DR	断开连接请求	√	√	√	√	√	1000	≤64 字节
DC	断开连接确认		√	√	√	√	1100	无
DT	数据	√	√	√	√	√	1111	协商好的长度以下
AK	数据确认		√	√	√	√	0110	无
ED	加速数据		√	√	√	√	0001	≤16 字节
EA	加速数据确认		√	√	√	√	0010	无
RJ	拒绝		√		√		0101	无
ER	错误	√	√	√	√	√	0111	无

以上各类 TPDU 解释如下：

- CR（Connection Request，连接请求）：向对端发送的面向连接的传输连接请求 TPDU。该类 TPDU 头长 7 个字节，整个 TPDU 长度可变。
- CC（Connection Confirm，连接确认）：向连接请求者发送的面向连接的传输连接确认 TPDU。该类 TPDU 头长 7 个字节，整个 TPDU 长度可变。
- DR（Disconnect Request，断开连接请求）：向对端发送的断开连接请求 TPDU。该类 TPDU 头长 7 个字节，整个 TPDU 长度可变。
- DC（Disconnect Confirm，断开连接确认）：向连接请求者发送的断开连接确认 TPDU。该类 TPDU 头长 6 个字节，整个 TPDU 长度可变。
- DT（Data，数据）：向对端发送的真正用户数据 TPDU。该类 TPDU 头长 3 个字节，整个 TPDU 长度可变。
- AK（Acknowledgement，数据确认）：向数据发送者发送的用户数据确认 TPDU，指出希望收到的下一个 TPDU 号。该类 TPDU 头长 5 个字节，整个 TPDU 长度可变。
- ED（Expedited Data，加速数据）：向对端发送的加速数据 TPDU。该类 TPDU 将优先普通用户数据 TPDU 发送，TPDU 头长 5 个字节，整个 TPDU 长度可变。
- EA（Expedited Data Acknowledgement，加速数据确认）：向加速数据发送者发送的加速数据确认 TPDU。该类 TPDU 将优先普通用户数据确认 TPDU 发送，TPDU 头长 5 个字节，整个 TPDU 长度可变。

- RT（Reject，拒绝）：向连接请求者发送的拒绝连接请求 TPDU。该类 TPDU 头长 5 个字节。
- ER（Error，错误）：向数据发送者发送的数据错误 TPDU，表示所收到的对应 TPDU 有错误。该类 TPDU 头长 5 个字节，整个 TPDU 长度可变。

具体来说，在 TPDU 头部后两部分要对应具体的传输层协议（也就是上节所说的 TP0～TP4）才有意义。但无论是哪种 TPDU，在发送端都是要自上而下传输的，到了网络层后同样要经过网络层协议的封装，形成分组，然后继续向下传输到数据链路层，封装成帧。TPDU 的逐层封装示意图如图 10-6 所示。

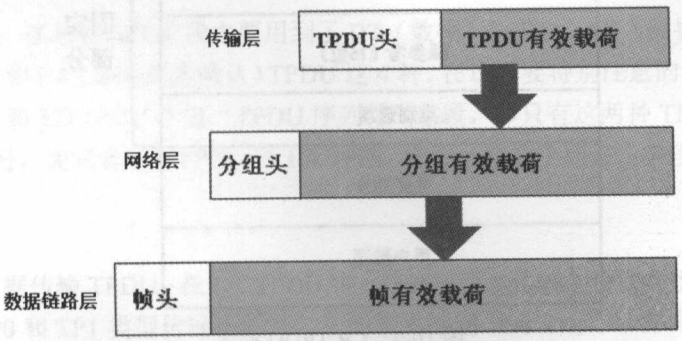


图 10-6 TPDU 封装示意图

TPDU 的“有效载荷”部分包含上层的用户数据。在每一个 TPDU 中，这部分的长度均受限制。如 CD/CC 类 TPDU 中该部分的长度不能超过 32 个字节；DR 类 TPDU 中该部分的长度不能超过 64 个字节；ED 类 TPDU 中该部分的长度为 1～16 个字节；DT 类 TPDU 中该部分的长度受连接建立时协商 TPDU 大小的限制。常见的 TPDU 参数参见表 10-2。

10.1.5 传输连接建立阶段的主要 TPDU

从本节开始要介绍上节说到的一些主要 TPDU 格式，看看数据传输的各个阶段使用了哪些主要 TPDU 以及它们所包含的基本内容是什么。本节介绍在传输连接建立阶段所用的主要 TPDU 格式。

从表 10-2 可知，在数据传输阶段主要用到了 CR（连接请求）TPDU 和 CC（连接确认）TPDU 这两种。

1. CR TPDU

CR TPDU 的基本格式如图 10-7 所示，其各字段的说明（仅说明“固定部分”）如下：

- 代码（Code）：标识 TPDU 类型，占 4 位，CR TPDU 的代码值为 1110。
- CDT（Credit Alocation）：信用量分配，表示能接收的 TPDU 序号的上限值（暗示了可接收的 TPDU 数量），占 4 位。CDT 类似于本章后面在介绍 TCP 数据段格式中所说到的“窗口大小”字段，对端通过对 CDT 字段值的识别来改变当前的发送窗口大小。TPDU 中的

CDT 字段与 TCP 数据段的“窗口大小”字段一样均用于流量控制，其基本原理类似于数据链路层和网络层的滑动窗口式流量控制。CR TPDU 中的 CDT 属于初始信用量，在传输连接建立时双方协商确定。

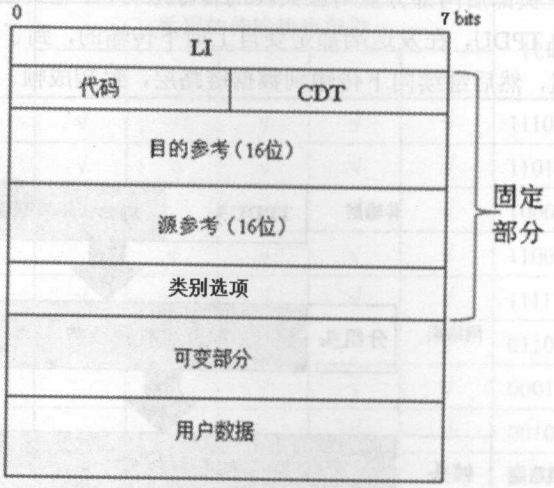


图 10-7　CR TPDU 格式

- 目的参考（DST Reference）：指该传输连接的目的端对等传输实体用来唯一标识这一连接的标识符，也就是对端传输连接序号，占 16 位。CR TPDU 中的“目的参考”为全 0，因为连接尚未建立（正在请求中），将由目的端选择。
- 源参考（Source Reference）：本地传输连接标识符，也就是本地传输连接的序列号，占 16 位。

【说明】在传输连接的每一端，源 TSAP 地址和目的 TSAP 地址写在 CR TPDU 头部中的可变部分。当目的端要对收到的 CR TPDU 表示确认并同意建立传输连接时就会发回 CC TPDU，其中也包含了“源参考”和“目的参考”这两个字段值。但此时 CC TPDU 中的“源参考”和“目的参考”都是针对发送 CC TPDU 的端点而言的，所以其中的“源参考”也就是在 CR TPDU 中以全 0 格式填充的“目的参考”。通过这样的两次握手过程双方就知道了各自的传输连接序号。

- 类别选项：占 8 位，高 4 位标明使用的传输层协议类型（TP0～TP4），低 4 位是可选项，说明在 DATA TPDU 和 AK TPDU 中的序号字段是正常的还是扩展的，正常为 0000，扩展为 0010。通常 TPDU 序号只有 7 位，但对于 TP2～TP4 传输层协议，序号可以扩展到 32 位。

协议和序号类型都是可协商的，但在协商过程中，服务质量只能降低而不允许提高。协商后的数值通过 CC TPDU 返回到源端，例如 CR 的发送端选择了 TP4（值为 1000），CC 的发送端可以将它降低为 TP2（0010）。但若 CR 的发送端选择的是 TP2，则 CC 的发送端不能将它改为 TP4。

CR TPDU 头部可变部分可以包括：TPDU 最大长度、源 TSAP ID、目的 TSAP ID、安全参数、

校验和、QoS 参数等，具体不作介绍了。

2. CC TPDU

CC TPDU 的格式与 CR TPDU 的一样（参见图 10-7），不同的只是 CC TPDU 中的“代码”字段为 1101，代表为 CC TPDU，同时“目的参考”字段不再是 16 位全 0，而是发起连接端的传输连接序列号，即 CR TPDU 中的“源参考号”，CC TPDU 中的“源参考号”为提供服务者的传输连接序号。

10.1.6 数据传输阶段的主要 TPDU

从表 10-2 可知，在数据传输阶段主要用到了 DT（数据）TPDU、AK（数据确认）TPDU、ED（加速数据）TPDU 和 EA（加速数据确认）TPDU 这 4 种。在这里要特别注意的一点是，所有 TPDU 中，只有 DT TPDU 和 ED TPDU 才有“TPDU 序列号”字段，即只有这两种 TPDU 才会进行编号，其他 TPDU 均不编号，而只有 AK TPDU 和 EA TPDU 才有“你的序列号”字段，其他 TPDU 均没有该字段。

1. DT TPDU

DT TPDU 是数据传输 TPDU。在 DT TPDU 中，选择使用的传输层协议类型不同对应的格式也不一样。当选择 TP0 和 TP1 类型传输协议时，它的格式如图 10-8 所示。

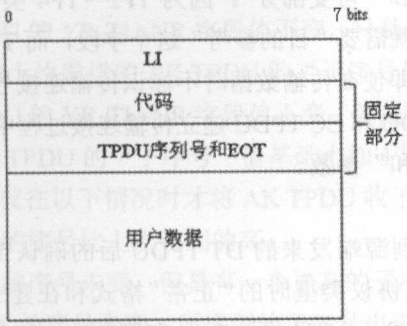


图 10-8 TP0 和 TP1 类型的 DT TPDU 格式

其各字段说明如下（仅说明“固定部分”）：

- 代码：标识 TPDU 类型，占 8 位，DT TPDU 的代码值为 1111 0000。
- TPDU 序列号和 EOT：占 8 位，高 7 位标识对应 DT TPDU 的序列号，最低 1 位（EOT 位）用来标识此 TPDU 是否是某个数据的最后一个数据段，等于 1 时表示是最后一个数据段。

当选择 TP2、TP3 和 TP4 类型传输协议时，它的格式要分两种情况：一是在建立传输连接时选择的不是 TP2～TP4 类型，此时的 TP2、TP3 和 TP4 类型 DT TPDU 格式如图 10-9 所示；二是如果在建立传输连接时也选择的是 TP2～TP4 类型，此时的 TP2、TP3 和 TP4 类型 DT TPDU 格式如图 10-10 所示。

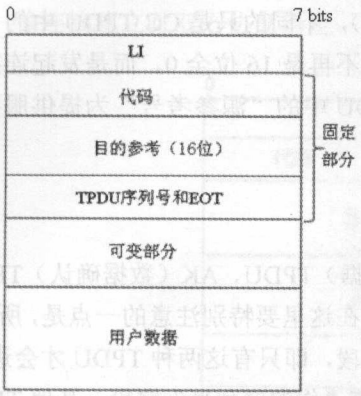


图 10-9　建立连接时未选择 TP2～TP4 类型时的 TP2～TP4 DT TPDU 格式

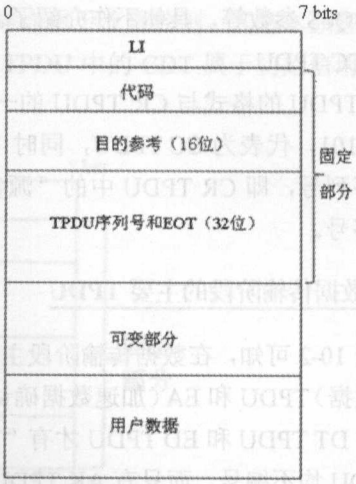


图 10-10　建立连接时已选择 TP2～TP4 类型时的 TP2～TP4 DT TPDU 格式

以上两种情况下 DT TPDU 格式的主要区别在于“TPDU 序列号和 EOT”字段的长度，图 10-9 中的为 8 位，而图 10-10 中的为 32 位。对比图 10-8 可以看出，在图 10-9 和图 10-10 所示的格式中，主要是多了一个“目的参考”和“可变部分”。因为 TP2～TP4 类型传输层协议支持多路复用了，需要标识各传输连接号，所以就需要“目的参考”这个字段，而 TP0～TP1 类型传输层协议是不支持多路传输连接复用的，所以即使在传输数据时不标识传输连接号也不会出现差错。“目的参考”字段占 16 位，是在通过 CR TPDU 和 CC TPDU 建立传输连接过程中已经获得的目的端传输连接号。在“可变部分”中包括“校验和”参数。

2. AK TPDU

AK TPDU 是接收端在收到源端发来的 DT TPDU 后的确认 TPDU。它的 TPDU 格式分在建立传输连接时没有选择具体的协议类型时的“正常”格式和在建立传输连接时已选择 TP1～TP4 类型传输协议时的“扩展”格式，分别如图 10-11 和图 10-12 所示。两者的主要区别在“代码”：CDT 和“你的 TPDU 序列号”这 3 个字段的宽度。其各个字段说明如下（仅说明“固定部分”）：

- 代码：标识 TPDU 类型，在正常情况下是占 4 位，为 0110；扩展情况下是占 8 位，为 0110 0000，代表为 AK TPDU。
- CDT：信用量分配，用于流量控制，在正常情况下是占 4 位，在扩展情况下是占 16 位，具体在本章前面已有介绍。
- 目的参考：占 16 位，是数据传输源端的传输连接序号。
- 你的 TPDU 序列号（YR-TU-NR）：Your-TPDU-Number，表明所期望接收的下一个 DT TPDU 的序号，这也隐含地表示对该序号以前的各 DT TPDU 均已加以确认，相当于本章后面将要介绍的 TCP 数据段中的“确认号”字段。在正常情况下是占 8 位，在扩展情况下是占 32 位。

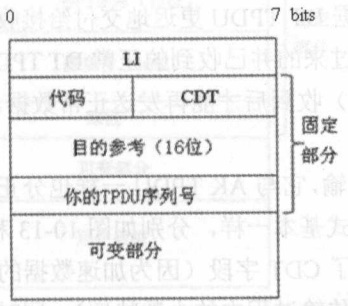


图 10-11　正常情况下的 AK TPDU 格式

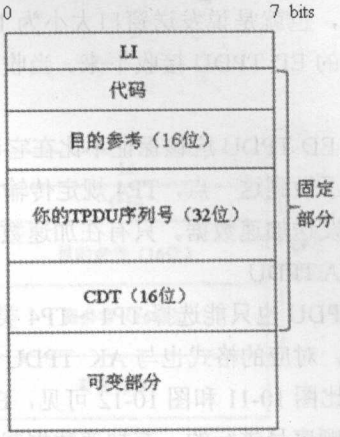


图 10-12　扩展情况下的 AK TPDU 格式

为了避免因 YR-TU-NR 字段位数不足而出现序列混乱现象，还可以利用 AK TPDU 中的“可变部分”参数的“子序号”参数（还包括“校验和”和“流量控制确认”参数）进行 TPDU 序号标识。子序号的使用规则如下：

- 当给一个具有更高序号（也就是 YR-TU-NR 字段值还未达到最大值）的 DT TPDU 发回 AK TPDU 时，应将“子序号”复位置 0，表示不采用“子序号”字段。
- 若本次发送的 AK TPDU 的 YR-TU-NR 字段值不变，只是信用量 CDT 字段值增大了，则“子序号”不变，即与上次发送的 AK TPDU 的“子序号”值相同。
- 若本次发送的 AK TPDU 的 YR-TU-NR 字段值不变，但是信用量 CDT 字段值减小了，则需要在上次发送的 AK TPDU 的“子序号”值基础上加 1。

在接收 AK TPDU 的一端，仅在以下情况时才将 AK TPDU 收下：

- 它所确认的 DT TPDU 的序号比上次收到的高。
- 它所确认的 DT TPDU 的序号未变，但具有一个更高的子序号。
- 它所确认的 DT TPDU 的序号未变，所携带的子序号也未变，但具有一个更大的信用量 CDT。

若同时收到若干个 AK TPDU，则按 YR-TU-NR 字段从小到大进行处理。当 YR-TU-NR 字段值相同时，按“子序号”字段值从小到大进行处理。如果“子序号”字段值也相同，则按信用量 CDT 字段值从小到大进行处理。

3. ED TPDU

ED TPDU 只能选择 TP1～TP4 类型传输协议进行传输，它与 DT TPDU 一样也分在建立连接时是否选择了 TP2～TP4 传输类型两种情况，对应的格式也与 DT TPDU 的两种情况下的格式完全一样（分别参见图 10-9 和图 10-10），不同的只是其中的“代码”值为 0001（建立传输连接时未选择 TP2～TP4 协议类型时）或者 0001 0000（建立传输连接时已选择 TP2～TP4 协议类型时）。

对于 ED TPDU 的流量控制比较简单，因为在任何时刻仅允许一个未被确认的 ED TPDU 在

通信途中，也就是说发送窗口大小为 1。在接收端，即使当正常数据的信用量的大小为 0 时也必须将收到的 ED TPDU 接收下来。当收到重复的 ED 时也要发回确认，不过要将重复的 ED TPDU 丢弃。

一个 ED TPDU 应该保证不比在它以后发出的正常数据 DT TPDU 更迟地交付给接收端的传输用户。为了做到这一点，TP4 规定传输实体将传输用户发过来的并已收到的正常 DT TPDU 暂停发送，而先发送加速数据。只有在加速数据确认（EA TPDU）收到后才能再发送正常数据。

4. EA TPDU

EA TPDU 也只能选择 TP1～TP4 类型传输协议进行传输，它与 AK TPDU 一样也分正常和扩展两种情况，对应的格式也与 AK TPDU 的两种情况下的格式基本一样，分别如图 10-13 和图 10-14 所示。对比图 10-11 和图 10-12 可见，EA TPDU 中只是少了 CDT 字段（因为加速数据的发送是不按 TPDU 顺序号进行的，有加速数据就立即发送，是数据传输过程中的少数特例），同时“代码”字段均为 0010 0000，代表是 EA TPDU。其他字段与 AK TPDU 中对应字段的作用相同。

【说明】数据发送的初始信用量分配规则为：对于流向目的端的数据，其初始信用量分配由 CC TPDU 提供；而对于流向源端的数据，其初始信用量分配是由 CR TPDU 提供的。以后的数据发送，信用量分配的大小是通过 AK TPDU 来分配的。

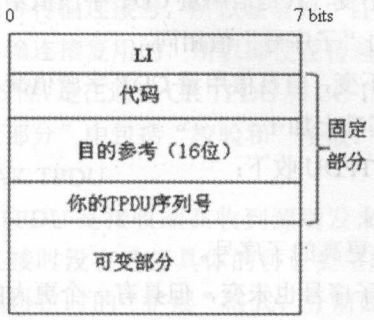


图 10-13　正常情况下的 EA TPDU 格式

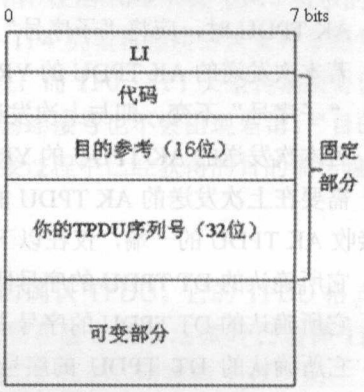


图 10-14　扩展情况下的 EA TPDU 格式

10.1.7　传输连接释放阶段的 TPDU

在连接释放阶段主要的 TPDU 是 DR TPDU 和 DC TPDU 这两种，前者是断开连接请求 TPDU，后者是断开连接确认 TPDU。

DR TPDU 和 DC TPDU 的格式分别如图 10-15 和图 10-16 所示。其中 DR TPDU“代码”字段的值为 1000 0000，DC TPDU“代码”字段的值为 1100 0000。DR TPDU 中的“原因”字段会显示断开连接的原因，其他字段参见前面对应小节的说明。

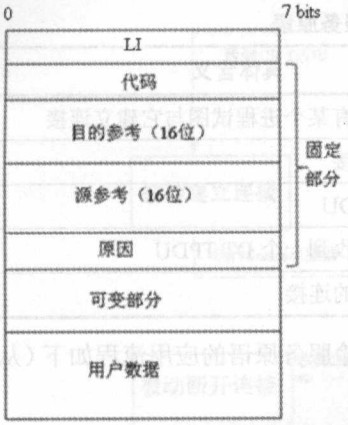


图 10-15 DR TPDU 格式

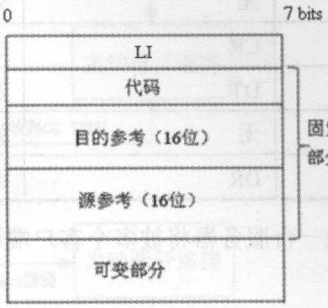


图 10-16 DC TPDU 格式

10.1.8 传输服务原语

“服务原语”（Service Primitive）是服务功能实现的具体过程描述，也就是要完成一项服务必须要经过的主要步骤，类似于本章后面介绍的 TCP/IP 体系结构中的 TCP 协议的“有限状态机”。一个服务通常由一组原语操作来描述，用户进程通过这些原语操作可以访问该服务。这些原语告诉该服务执行某个动作或者将某个实体所执行的动作报告给用户。如果协议栈位于操作系统中（大多数情况是这样的），则这些服务原语通常是一些系统调用。这些系统调用会进入到内核模式，然后在内核模式中控制该机器，让操作系统发送必要的分组。到底哪些原语可以使用取决于所提供的服务，针对面向连接的服务的原语与针对无连接的服务的原语是不同的。在 OSI/RM 体系结构中，服务原语每层都有，并不只是传输层才有。

传输服务类似于网络服务，但两者存在显著的区别。网络服务倾向于将实际网络提供的服务模型化，由于网络内部问题，实际的网络可能会丢失数据分组，所以网络服务并不十分可靠。相反，面向连接的传输服务则是可靠的，虽然实际的传输服务并不是毫无错误，但在不可靠的网络服务之上提供可靠的传输服务正是传输层要实现的目标。

1. 一个简单的传输服务原语模型

在正式介绍传输层服务原语前，我们先来通过一个最基本、最简单的传输服务原语模型来理解传输服务原语的含义。这个简单的传输服务模型就仅包括：监听（Listen）、连接（Connect）、发送（Send）、接收（Receive）、断开连接（Disconnect）。它代表了一个 C/S（客户/服务器）模式的数据传输服务基本过程，这也是面向连接的传输层服务的 5 个基本过程。这 5 个基本服务原语所发送的分组和所代表的含义如表 10-3 所示。有关 TPDU 请参见前面各小节的介绍。

表 10-3　5 种基本的传输服务原语

服务原语	可发送的 TPDU	具体含义
Listen	无	处于阻塞状态，直到有某个进程试图与它建立连接
Connect	CR	主动请求建立一个连接
Send	DT	发送用户数据 DT TPDU
Receive	无	处于阻塞状态，直到收到一个 DT TPDU
Disconnect	DR	请求释放一个已建立的连接

现假设一台服务器将被多个客户端访问，这些基本传输服务原语的应用流程如下（从最初状态开始）：

（1）一开始，服务器是调用 Listen 原语，使其处于监听状态，直到有客户来请求连接。

（2）当一个客户希望与该服务器进行会话时，它调用 Connect 原语，向服务器发送一个请求建立传输连接的 CR TPDU。

（3）当该 CR TPDU 到达服务器时，服务器传输服务实体执行状态检查，看服务器是否正处于监听状态。如果是，则解除监听，并给客户送回一条确认连接请求的 CC TPDU。

（4）当这个 CC TPDU 到达客户端后，客户端获知服务器同意建立传输连接了，随后便通知客户端应用进程开始建立传输连接。

（5）传输连接建议好后就可以正式发送 DT TPDU 了。任何一方都可以调用 Receive 原语，以等待另一方调用 Send 原语，发送数据。当一方接收到另一方发来的 DT TPDU 时，接收端会发送一个数据接收确认 DA TPDU。

（6）当不再需要一个连接时，传输用户必须将它释放，以便使两个传输实体内部的缓存空间有机会被重新使用。

释放连接有两种形式：对称释放和非对称释放。在非对称释放中，任何一方都可以调用 Disconnect 原语，在本端传输实体向对端发送一个 DR TPDU 后双向传输连接都将被释放。在对称方式中，连接的两个方向彼此独立，每个方向需要单独被释放。当一方调用 Disconnect 原语时，这意味着它不再需要发送数据了，但是它仍然希望能接收对方发送过来的数据。在这种方式中，只有当双方都调用了 Disconnect 原语，各自向对方发送 DR TPDU 后一个传输连接才可能被真正释放出来。

图 10-17 给出了全用这些简单传输服务原语来建立和释放连接的状态图。每一个状态的出现都是一种事件触发，或者是由本地的传输用户执行了一个原语，或者是接收到了一个 TPDU 分组。不过要注意，此处为了简单起见，假设了每一个 TPDU 都是单独确认的，而且也采用了对称释放模型，由客户先释放连接。图中实线部分显示了客户端的状态序列，而虚线部分则显示了服务器的状态序列。

2. ISO 规范中的传输服务原语

在 OSI/RM 体系结构的 ISO 规范中，各层服务原语分为以下 4 种类型：

- 请求（Request）：用户利用它要求服务提供者提供某项服务，如建立连接或发送数据等。

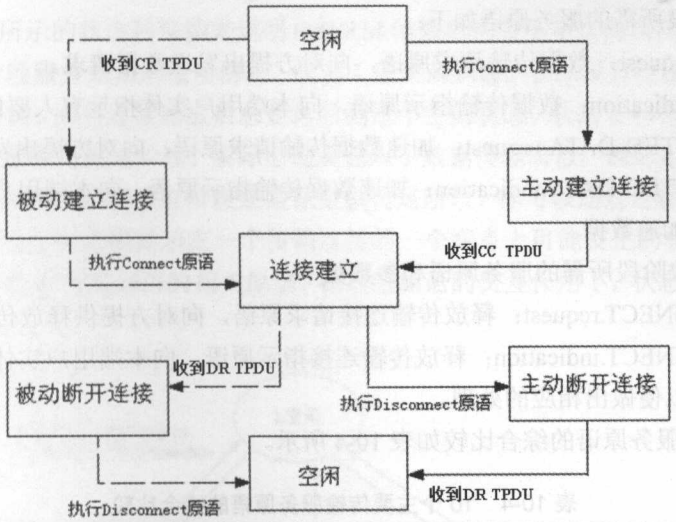


图 10-17 一个简单传输服务的原语应用示例

- 指示（Indication）：服务提供者执行请求原语后，用指示原语通知本端用户实体，告知有人想要与之建立连接或发送数据等。
- 响应（Response）：收到指示原语后，利用响应原语向对方作出反应。例如同意或不同意建立连接等。
- 确认（Confirm）：请求方可以通过接收确认原语来获悉对方是否同意接受请求。

在这 4 个服务原语中，Request 和 Response 由请求服务用户调用，而 Indication 和 Confirm 则是由服务提供者调用。各类服务原语均可以携带参数，如请求原语的参数可能指明它要与哪台机器连接、需要什么服务类别等；指示原语的参数可能包含呼叫者的标识、需要服务的类别等；响应原语的参数可能包括同意或不同意连接的关键字；在确认原语中，则可能对某些参数给出协商值，如最大数据吞吐量等。

在 OSI/RM 传输层中主要包括 10 种传输服务原语，分布在传输连接建立、数据发送和传输连接断开这 3 个主要阶段中。其中在传输连接建立阶段所需的 4 个服务原语及参数如下（小圆点后面部分代表对应的传输服务原语类型，下同）：

- T-CONNECT.request：传输连接请求原语，客户端通过 CR TPDU 向服务器请求与对方建立一条传输连接。
- T-CONNECT.indication：传输连接指示原语，服务器向自己的用户实体（用户应用进程）指示有客户要求与自己建立传输连接。
- T-CONNECT.response：传输连接响应原语，服务器通过 CC TPDU 向客户端发送一条传输连接建立响应。
- T-CONNECT.confirm：传输连接确认原语，客户端收到服务器的传输连接响应后向本端用户实体（用户应用进程）确认服务器是否已接受连接请求。

在数据传送阶段所需的服务原语如下：

- T-DATA.request：数据传输请求原语，向对方提出发送数据请求。
- T-DATA.indication：数据传输指示原语，向本端用户实体指示有人要向自己发送数据。
- T-EXPEDITED-DATA.request：加速数据传输请求原语，向对方提出发送回还数据请求。
- T-EXPEDITED-DATA.indication：加速数据传输指示原语，向本端用户实体指示有人要向自己发送加速数据。

在传输连接释放阶段所需的服务原语和参数如下：

- T-DISCONNECT.request：释放传输连接请求原语，向对方提供释放传输连接请求。
- T-DISCONNECT.indication：释放传输连接指示原语，向本端用户实体指示对端要释放传输连接，以便做出相应的处理。

以上 10 个传输服务原语的综合比较如表 10-4 所示。

表 10-4　10 个主要传输服务原语的综合比较

阶段	服务	使用的服务原语	参数
连接建立	建立传输连接	T_CONNECT.request T_CONNECT.indication	被叫地址、主叫地址、加速数据选择、服务质量、传输服务用户数据
		T_CONNECT.response T_CONNECT.confirm	服务质量、响应地址、加速数据选择、传输服务用户数据
数据传送	正常数据传送	T_DATA.request T_DATA.indication	传输服务用户数据
	加速数据传送	T_EXPEDITED_DATA.request T_EXPEDITED_DATA.indication	传输服务用户数据
连接释放	释放传输连接	T_DISCONNECT.request	传输服务用户数据
		T_DISCONNECT.indication	断开连接原因、传输服务用户数据

表中“主叫地址”参数是指请求传输连接的 TSAP 地址；“被叫地址”参数是指与之建立传输连接的 TSAP 地址；“响应地址”参数等于被叫 TSAP 地址；“加速数据选择”参数是指在传输连接上是否可以使用加速数据服务（如果声明没有这项服务，它就不能在传输连接上使用）；“传输服务用户数据”参数是在传输用户之间传送的传输服务数据，传输服务提供者不进行修改，其长度为 1～32 字节；“服务质量”参数包括传输服务用户的一些要求，即 QoS 参数。

无连接的传输服务只有两个原语，如表 10-5 所示。

表 10-5　无连接传输服务原语

原语	参数
T_UNIT_DATA.request	主叫地址、被叫地址、服务质量、传输服务数据
T_UNIT_DATA.indication	主叫地址、被叫地址、服务质量、传输服务数据

可用如图 10-18 所示的状态转移图来说明 OSI/RM 体系结构中所定义的面向连接传输服务在连接建立和释放两个阶段服务原语的使用情况。状态 1 是空闲状态，表示没有传输连接。每次连接的建立都从这一状态开始，而当连接释放后总是再回到这个空闲状态。状态 2 和状态 3 都属于连接建立阶段。状态 4 是数据传送就绪阶段，实际上也就进入了数据传输阶段，调用相应的原语后就可以进行数据传输了。无论是在连接建立阶段还是在数据传送阶段，都可以通过连接释放原语返回到原始的空闲状态。这个状态转移图指示在一个传输连接的一个端点上可能发生的状态序列，箭头旁边注明的是对应状态转移时所需调用的相关原语。在这些原语的交互作用下，状态沿箭头转移到下一个状态。

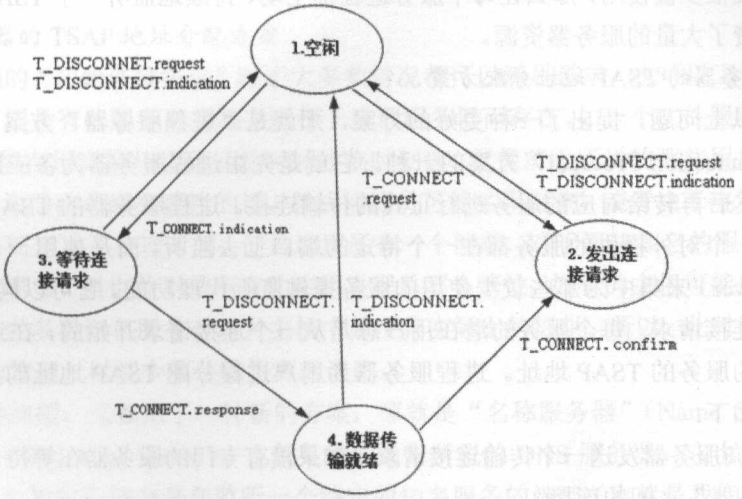


图 10-18　传输连接建立和释放阶段的一端主机状态转移示意图

10.2　传输层服务功能

从上节中的介绍我们知道，传输层要完成端到端的透明数据传输至少要分两大步骤：一是虚拟传输连接的建立（此处仍仅针对面向连接的传输层协议）；二是源端用户数据沿着传输连接传送到目的端。在整个数据传输过程中，传输层服务需要完成以下 8 个方面的基本功能：传输层寻址、传输连接建立、数据传输、传输连接释放、流量控制、拥塞控制、多路复用和解复用、崩溃恢复。这是本章的重点与难点，请大家着重理解这些功能的实现原理。

10.2.1　传输层寻址方案

在本章前面就已经说到，虽然网络层把数据分组从源主机传到了目的主机，但是这并不代表数据的传输过程就全部完成了，因为数据分组还没有真正地交付给相应的应用进程，此时应用进程才是数据通信主体。而我们又知道，同一时间的通信双方可能存在许多并发应用进程，那么源端来的

数据分组该交给哪个应用进程呢？这就涉及到传输层的寻址问题了。那就是在本章前面就已经介绍的 TSAP（传输服务访问点）。它代表的是一个传输层端口，因为传输层是以一个端对应一个应用进程来表示的。

应用进程（包括客户端和服务器端）可以将自己关联到一个 TSAP 上，以便与远程的 TSAP 建立连接。但是我们知道，只有少数常见的服务（如 WWW、FTP、POP、SMTP、DNS、DHCP 等）是固定分配了 TSAP 地址——端口的，绝大多数服务并没有与固定的端口进行绑定，所以这些服务的发送端就需要接收端对应用户进程所关联的 TSAP 地址。那么发送主机的用户进程如何知道接收主机对应用户进程所关联的 TSAP 地址呢？还有，在一台服务器上有大量潜在的服务进程，但是大多数服务进程又很少被使用，那么让每个服务进程都主动、持续地监听一个 TSAP 地址显然是非常不明智的，浪费了大量的服务器资源。

1. 进程服务器的 TSAP 地址分配方案

为了解决以上问题，提出了一种更好的方案，那就是“进程服务器”方案，是“初始连接协议”（Initial Connection Protocol）方案的一种。它就是先由进程服务器为各应用进程集中建立初始传输连接，然后再转给对应的服务进行正式的传输连接。进程服务器的 TSAP 地址分配方案的做法不是让每一个对外提供的服务都在一个特定的端口上去监听，而是使用一个特殊的进程服务器（Process Server）来集中为那些较少使用的服务提供监听代理功能。它可以同时监听一组端口，以等待外来的连接请求。每个服务的潜在用户总是从一个连接请求开始的，在连接请求中指定了这些用户想要的服务的 TSAP 地址。进程服务器为用户进程分配 TSAP 地址的过程如图 10-19 所示，具体解释如下：

（1）用户向服务器发送一个传输连接请求，如果没有专门的服务器在等待，则它们会得到一个与“进程服务器”之间的连接。

（2）进程服务器接到了用户的传输连接请求后，装载对应的用户进程，启动与该请求对应的服务。

（3）进程服务器为该用户进程分配一个临时的 TSAP 地址，这样用户与对应的服务器进行初始连接，允许对应的服务继承进程服务器与用户之间已有的连接。然后对应的服务执行用户请求的工作，而进程服务器则回去继续监听新的请求。

打个比方，某公司中有多个产品营销部门，每个部门设立了一个专门的售后服务热线电话。但其中有些部门的热线电话比较忙，经常有用户打电话进来咨询或投诉，而有些部门的热线电话打进来的人比较少。这时如果仍为每个部门专门安排一个人负责热线电话的话，显然对于那些用户参与比较少的热线电话来说这是一种浪费。这时我们肯定会想到，把这些比较少用户参与的部门热线电话用一个总线电话来转接，则这时只需要一个专门的接线员来管理总机。当有人打这个总线电话时，总机接线员询问所咨询的内容，然后再把电话转接到对应的部门分机，由对应的部门接线员进行处理，总机接线员又继续等待下一个用户电话，而这些部门的热线电话接线员平时可以进行其他工作，不必总在那里等待用户的来电，节省了资源。

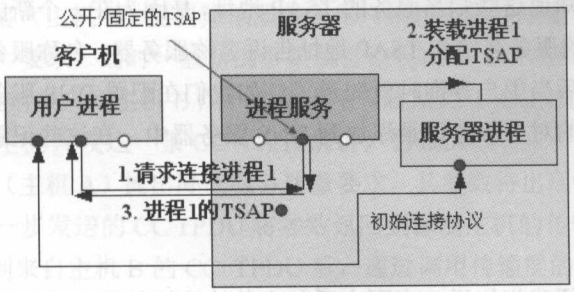


图 10-19 进程服务器为用户进行分配 TSAP 地址的基本流程

2. 名称服务器的 TSAP 地址分配方案

虽然以上介绍的“初始连接协议”对于大多数情况都可以满足需求，也可以工作得很好，但是在一些特殊情况下还是有些服务必须独立运行于进程服务器而存在。如一个文件服务器需要在特殊的硬件上运行，不能在用户要与它通话的时候才临时创建。就像前面所说的那些用户参与比较多的部门热线电话，如果仍由总线来转接的话，总线接线员可能忙不过来，或者由于电话过多而电话线路总处于占线状态而延误了处理时间。但是这里还有一个问题，对于那些知名的服务，在用户主机应用进程中仍然非常多，但也并不是所有的应用进程总那么繁忙，如有的主机可能仅担当少数几种服务器角色，而绝大多数知名服务在这台主机中并没有实际上的提供。所以，此时仍需要对这些知名服务提供集中的管理，以减少服务器资源的浪费。

为了解决这种问题，又推出了一种新的方案，那就是“名称服务器”（Name Server），有时也称“目录服务器”（Directory Server），也就是我们通常所说的 DNS 服务器。它也是“初始连接协议”方案的一种。名称服务器总是在监听一个特定的知名服务的端口（也就是我们通常所说的“常规端口”）。在这种方案中，为了找到与给定的服务名字相对应的 TSAP 地址（端口），用户需要与名称服务器建立一个连接，然后用户发送一条消息指定它想要的服务名字，名称服务器则发回相应的 TSAP 地址。随后，用户将它与名称服务器之间的连接释放掉，再与期望的服务建立一个新的连接。名称服务器为用户进程分配 TSAP 地址的整体流程与上面介绍的进程服务器为用户进程分配 TSAP 地址的流程基本类似，如图 10-20 所示，只是名称服务器效率更高。

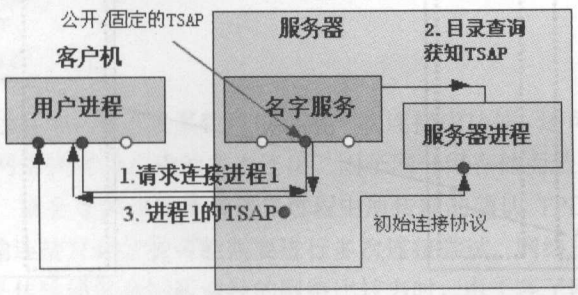


图 10-20 名称服务器为用户进程分配 TSAP 地址的基本流程

名称服务器之所以知道这些知名服务的 TSAP 地址，是因为在一个新的服务被创建时必须向名称服务器注册，并把它的服务名字和 TSAP 地址告诉名称服务器。名称服务器将把这份信息记录到它的内部数据库中，以后当用户查询时就知道了。如我们在配置 DNS 服务器时就会把这台机器上所配置的各服务器角色的对应服务记录添加到 DNS 服务器中，在这些记录中就会记录对应的服务器名称、所用端口信息。

10.2.2 传输连接建立

通过本章前面的介绍我们知道，表面上看起来传输连接的建立过程很简单，用户端只需调用 T_CONNECT.request 和 T_CONNECT.confirm 原语，服务器端只需调用 T_CONNECT.indication 和 T_CONNECT.response 原语即可。其实传输连接的建立非常繁锁，因为它受到网络延迟、子网分组存储、重复分组接收等诸多因素影响。下面先看一下正常情况下基本的传输连接过程。

1. 正常传输连接的建立

正常情况下，传输连接的建立过程比较简单，所用到的传输服务原语就是前面所说到的那 4 个，所用到的 TPDU 主要有 CR TPDU 和 CC TPDU 这两个。基本的传输连接建立过程如图 10-21 所示，过程描述如下：

（1）客户端主机 A 用户实体中的对应用户进程调用传输层的 T_CONNECT.request 服务原语，由所使用的传输层协议向服务器端主机 B 发送一条 CR（连接请求）TPDU，请求建立一条传输连接。

【说明】在 OSI/RM 体系结构中，用户实体位于“会话层”中，而 TCP/IP 协议体系结构中，用户实体位于“应用层”中，下同。

（2）主机 B 传输层在收到来自主机 A 的这条连接请求 TPDU 后，通过调用 T_CONNECT.indication 服务原语告诉自己的用户实体中的对应用户进程：有用户想与自己建立传输连接。

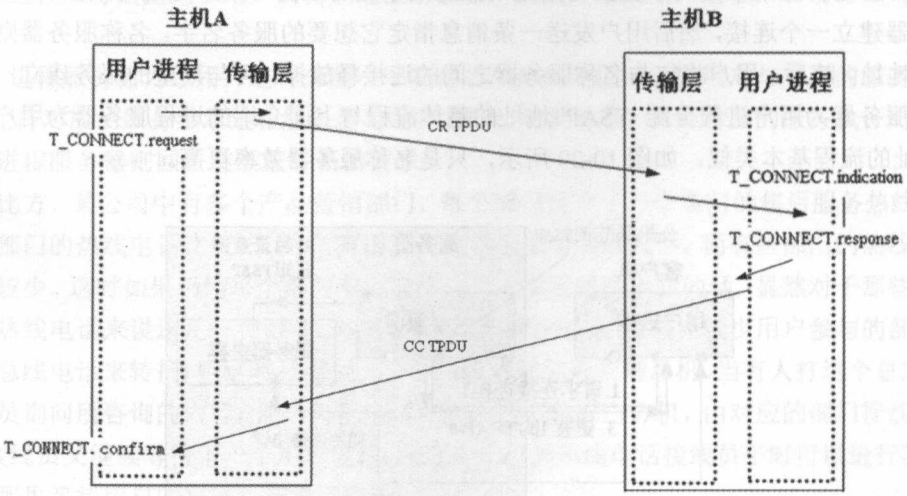


图 10-21 传输连接建立的基本流程

如果传输服务提供者（主机 B）提出降低服务质量（如较小的吞吐量、较长的时延、较高的差错率、较低的优先级等）要求，这时一些新的参数就会出现在 T_CONNECT.indication 原语中。

（3）如果接受连接请求，则主机 B 用户实体调用传输层的 T_CONNECT.response 服务原语，然后通过传输层协议向主机 A 发送一条 CC（连接确认）TPDU。

如果传输服务用户（主机 A）提出降低服务质量要求，其参数将出现在 T_CONNECT.response 原语中。最后，通过下一步发送的 CC TPDU 将参数返回到源端主机的传输用户。

（4）主机 A 在收到来自主机 B 的 CC TPDU 后，通过调用传输层的 T_CONNECT.confirm 服务原语向自己的用户实体中的源用户进程确认连接已建立，正式进入数据发送阶段。

从以上的步骤可以看出，传输连接的建立过程很简单，但实际上传输连接的建立远不是这样，甚至可以用非常繁锁来形容，主要体现在因网络拥塞而出现多次连接，因网络中存在延迟的重复分组而造成重复连接。

2. 传输连接拒绝

需要注意的是，“传输连接拒绝”不是“传输连接释放”，它是在建立连接过程中被对端拒绝，此时传输连接根本还没有建立起来，在建立过程中就被中断了。

在以上“传输连接建立”过程中的第 3 步，如果连接的目的主机 B（服务提供者）传输层可能因源端的服务质量太差或者自己的线路很忙而要拒绝主机 A（服务请求者）的传输连接请求，这时就可以通过发送 RJ TPDU 来实现了，具体过程如下：

（1）当来自主机 A 的 CR TPDU 到了主机 B 传输层后，通过调用自己传输层的 T_CONNECT.indication 原语向对应的用户进程说明有用户需要与它建立传输连接，同时会通过对 CR TPDU 的分析得出源端的服务质量水平。

（2）如果主机 B 应用进程认为源端的服务质量水平太差，或者应用层目前工作实在太繁忙，不接受此次的传输连接请求，则通知传输层向主机 A 发送 RJ TPDU，表示拒绝此次传输连接。

在“传输连接拒绝”过程中的整个服务原语调用和 TPDU 发送情况如图 10-22 所示，与图 10-21 所示的“传输连接建立”过程相比，唯一的区别就是主机 B 向主机 A 发送的不再是 CC TPDU，而是 RJ（拒绝连接）TPDU。当然在主机 B 传输层向自己的应用层提交的 T_CONNECT.indication 原语中的内容也会有所不同，会表示出不愿意接受连接的意愿，尽管最终决定是否接受连接的是应用层的用户进程而不是传输层。

10.2.3 重复传输连接的解决方法

在传输连接建立的过程中，除了大多数可以正常一次连接成功外，还可能因各种原因出现一些非正常连接现象，如因网络拥塞而造成的多次连接、因子网分组存储而造成的重复连接等。

如果网络非常拥塞，就会导致传输连接建立过程中所传输的确认 TPDU 分组都无法及时回到发送端，这样一来，传输连接发起方就可能需要进行多次连接尝试。网络延迟还可能因为“存储一转发”机制引起，特别是传输通信两端所途经的网络比较多时，由于各子网的分组存储功能，导致一些 TPDU 分组没有及时到达，最终造成传输连接发起方会多次重试。但问题远不是重试这么简

单，因为这些延迟的分组可能在过一段时间后到达了，于是又会请求建立新的连接，这就出现了重复连接的问题。

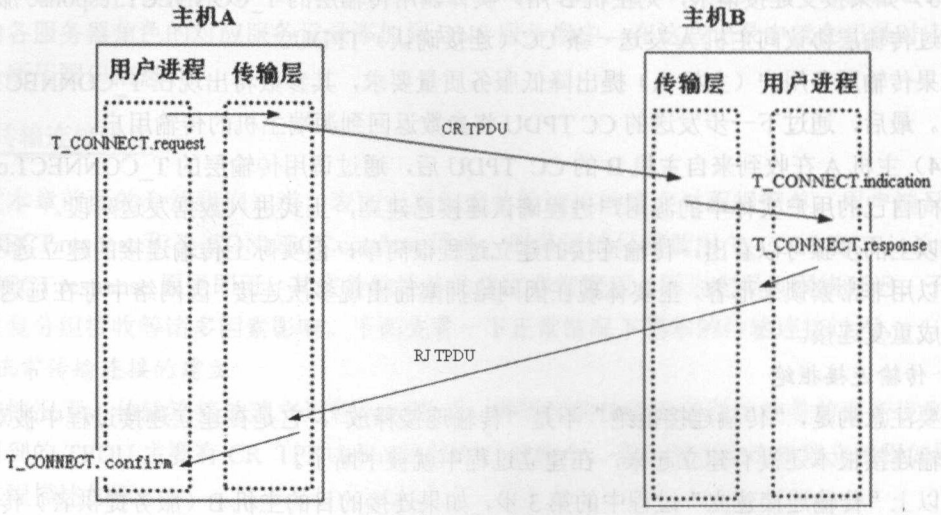


图 10-22　传输连接请求拒绝的基本流程

对于重复的传输连接问题可以说是没有一种特别有效的解决方法。通常是采取非重复 TSAP、超时连接表、分组的 TTL 机制和三次握手机制等方法进行尽量避免。

（1）非重复 TSAP。

这一方法是系统为每次连接赋予一个新的 TSAP 地址，当连接被释放时将此 TSAP 废弃。这样一来，就不会出现重复连接的现象。但这种解决方法不适用于那些知名服务的连接，因为像 WWW、FTP、POP 之类的传输连接是固定使用那些知名的端口进行的，不可能每次都变。

（2）过时连接表。

这一方法是为每个连接分配连接标识符（也就是传输连接序列号），每新建一个传输连接序列号就增加 1，并且存入到每个 TPDU（也包括请求建立连接的 TPDU）中。当每个连接被释放时，将此连接信息存入过时连接表，然后在每个新的连接请求到达时，根据标识符核对过时连接表，如是已过时的序列号则视为重复连接。但在这种方法中需要每个传输实体无限期地维护一张历史记录表，不适应突变情况，如机器重启、系统崩溃等。

（3）分组的 TTL 机制。

这一方法是先定义和计数每个分组的生存时间 TTL，超时未达目的地，视为陈旧分组，被抛弃。同样也存在一些问题，如消除重复连接请求需要依赖子网完成、网络层分组传输本身的不可靠性等。

（4）三次握手机制。

所谓“三次握手”是指在建立连接时，发送端发送 CR TPDU 请求建立一个连接（第一次握手）；

接收端收到 CR TPDU 后发送一个 CC TPDU 进行应答（第二次握手），在应答分组中宣告了自己的初始连接序列号；发送端收到应答分组后发送第一个 DT TPDU，对接收端初始序列号确认（第三次握手）。这样，一个连接才能建立起来，并且双方可以使用不同的起始序号。整个三次握手机制如图 10-23 所示。

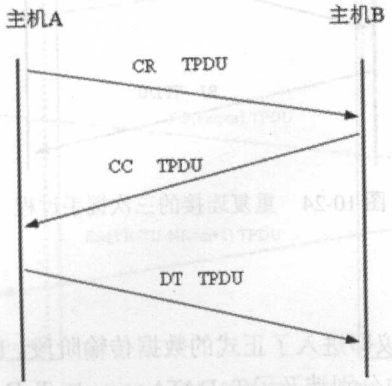


图 10-23 正常情况下的传输连接三次握手过程

简单描述如下（在 TCP/IP 协议体系结构中的 TCP 传输连接建立也是采用“三次握手机制”的，具体将在本章后面介绍）：在正常情况下，主机 A 向主机 B 发送一个“源参考”为 i（“目的参考”为 0）的 CR TPDU 请求建立连接；主机 B 收到 CR TPDU 后发送一个“源参考”为 j（“目的参考”为 i）的 CC TPDU 应答分组，对主机 A 发来的连接请求进行确认；主机 A 就从收到的 CC TPDU 的“源参考”字段中获知了主机 B 的传输连接参考，把它作为“目的参考”字段值发送第一个携带少量数据的 DT TPDU（其中包括了 DT TPDU 的序列号），同时对主机 B 选择的初始连接序号进行确认。

其实三次握手连接的原理很简单，就像我们平时聊天一样，先是由发起人说“Hello”（打招呼通常预示着要向对方发话），当对方看到这个招呼后如果确认此时可以与发起方聊天，则也会发一个招呼语“Hello”，这里有两重意思，一方面是证明他已收到发起方的招呼，另一方面也证明目前自己可以与发起方聊天，发起方在收到对方的回应招呼后可以再次选择，如果恰在此时又有其他事不能与对方聊了，当然可以放弃了，如果认为条件成熟，便会说“OK，Begin”（好的，开始吧），接下来就是正式的聊天了。

在出现延迟的重复 CR TPDU 的情况下，在网络中间延迟发送的一个源参考为 i 的重复 CR TPDU 到达主机 B 后，请求在已释放的连接上建立连接，主机 B 发送一个 CC TPDU 应答分组对原来源参考为 i 的请求进行应答，同时在“源参考”字段中宣告自己的传输连接号为 j；主机 A 通过检查可知主机 B 的应答分组中确认的源参考为 i 的连接原来已经确认，便认为该应答是过时的，并向主机 B 发送一个 RJ TPDU 拒绝连接，整个过程如图 10-24 所示。这样，通过三次握手法可以有效地克服延迟的重复 CR TPDU 分组的负面影响，提高了建立连接的安全性和可靠性。

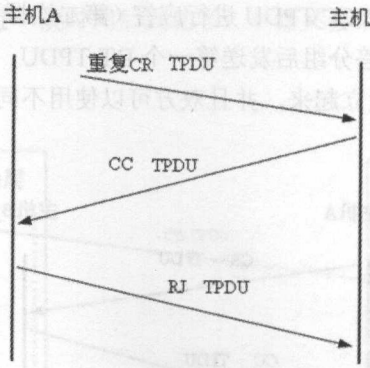


图 10-24　重复连接的三次握手过程

10.2.4　数据传输

一旦传输连接建立，传输协议即进入了正式的数据传输阶段。ISO 规定了正常数据传输和加速数据传输两种类型的服务。在其中分别涉及了 T_DATA.request、T_DATA.indication、T_EXPEDITED_DATA.request 和 T_EXPEDITED_DATA.indication 这 4 个传输服务原语，以及 DT TPDU、AK TPDU、ED TPDU、EA TPDU 这 4 个 TPDU。图 10-25 表示了两种数据传输时的服务原语和 TPDU 的使用情况，具体过程描述如下：

（1）当传输连接建立好后，需要发送普通数据时，主机 A 的用户实体调用 T_DATA.request 传输服务原语（发送加速数据时，调用 T_EXPEDITED_DATA.request 传输服务原语），指示传输层协议开始进行数据发送。普通数据是以 DT TPDU 进行发送的（其中“TPDU 序列号”字段值假设为 i），加速数据是以 ED TPDU 进行发送的（其中“TPDU 序列号”字段值假设为 m）。

【说明】如果所发送的数据是经过分段的，每个数据段 TPDU 都有子序列号（在“可变部分”中），在最后一个数据段中的 EOT（传输结束）置 1，如果不是最后一个数据段，则该位置 0。

（2）当数据到达主机 B 传输层时，普通数据通过 T_DATA.indication 服务原语（加速数据通过 T_EXPEDITED_DATA.indication 服务原语）向用户实体指示有数据到达。

（3）如果收到的是普通数据，主机 B 发送 AK TPDU 进行确认，其中有“你的 TPDU 序列号”（YR-TU-NR）字段为希望收到的下一个数据或加速数据 TPDU 的序列号，等于 i+1，但没有“TPDU 序列号”字段值；如果收到的是加速数据，则主机 B 发送 EA TPDU 进行确认，其中有“你的 TPDU 序列号”（YR-TU-NR）字段为希望收到的下一个数据或加速数据 TPDU 的序列号，等于 m+1。

10.2.5　传输连接释放

传输连接的释放分为“对称释放”和“非对称释放”两种，“对称释放”是各自独立发出释放连接请求，收到对方的释放确认之后才可释放连接；“非对称释放”是发送释放请求后单方终止连接，但这样做有可能丢失对方发送的数据。

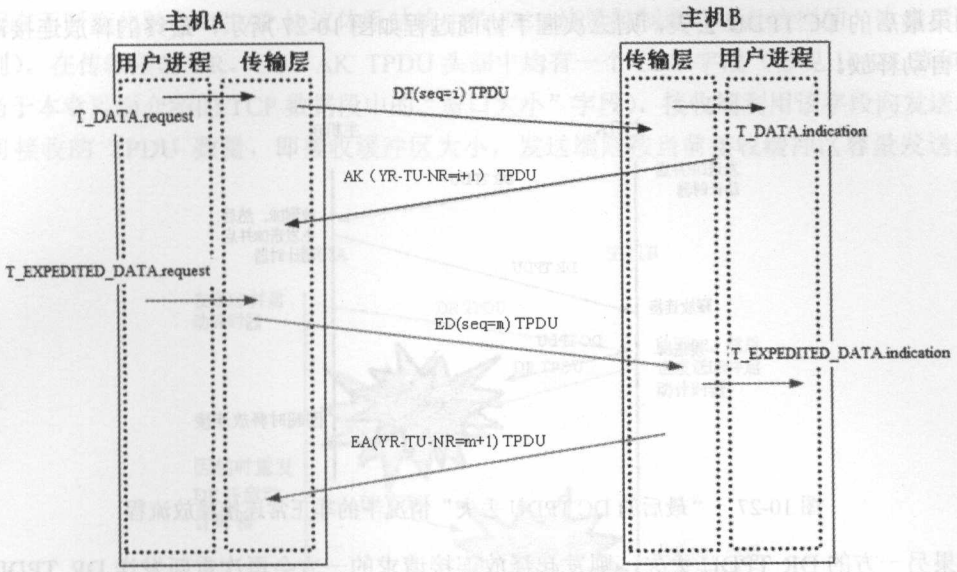


图 10-25　数据传输基本流程

OSI/RM 的传输连接释放也是采用三次握手协商方式（在 TCP/IP 协议体系结构中的 TCP 传输连接释放是采用“四次握手机制”，具体将在本章后面介绍），同样也分为“正常释放连接”和“非正常释放连接”两种情况。“非正常释放连接”又包括以下 3 种情况：最后的确认 TPDU 丢失、应答 TPDU 丢失、应答及后续释放请求 DR 丢失。本节以对称释放方式进行介绍。

正常情况下的三次握手连接释放过程如图 10-26 所示。图中省略了释放连接请求方用户实体调用的 T_DISCONNECT.request 传输服务原语和释放连接接受方用户实体调用的 T_DISCONNECT.indication 服务原语。所使用的 TPDU 有两类：用于请求释放连接的 DR TPDU 和用于确认释放连接的 DC TPDU。

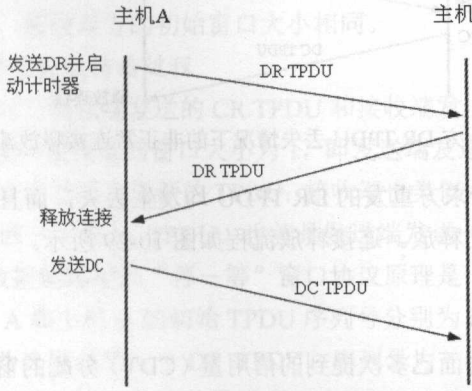


图 10-26　正常情况下的三次握手连接释放流程

如果最后的 DC TPDU 丢失，则三次握手协商过程如图 10-27 所示，最终的释放连接请求会因超时而自动释放。

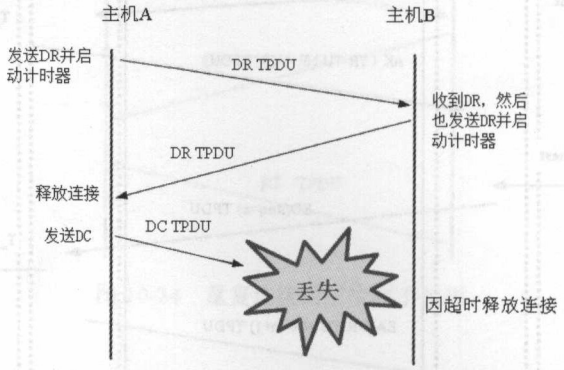


图 10-27　“最后的 DC TPDU 丢失”情况下的非正常连接释放流程

如果另一方的 DR TPDU 丢失，则发起释放连接请求的一方会再次重新发送 DR TPDU，请求释放连接。连接释放流程如图 10-28 所示，在请求方重发 DR TPDU 后释放连接。

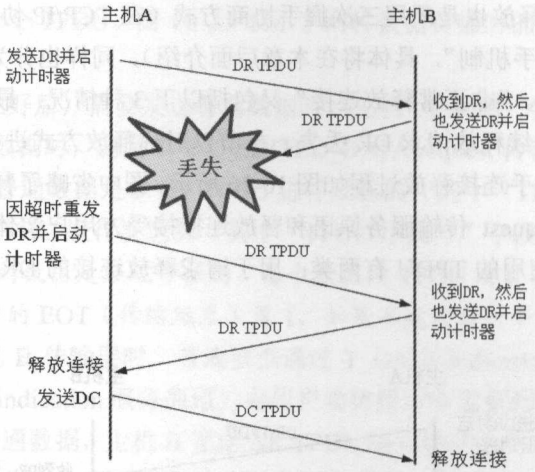


图 10-28　另一方 DR TPDU 丢失情况下的非正常连接释放流程

如果另一方的 DR TPDU 和请求方重发的 DR TPDU 均发生丢失，而且重发多次仍不能成功，则最后因传输超时而导致连接自动释放。连接释放流程如图 10-29 所示。

10.2.6　流量控制

传输层协议采用的是在本章前面已多次提到的信用量（CDT）分配的窗口机制，不同于数据链路层协议流量控制方案中的窗口尺寸固定的滑动窗口机制，因为传输层中的 CDT 值是随着数据的

传输过程在不断变化的（TCP/IP 协议体系结构中的 TCP 流量控制原理也是这样的，本章后面将具体介绍到）。在传输层的 CR、CC、AK TPDU 头部中均有一个 CDT 字段（参见 10.1.5 节和 10.1.6 节，相当于本章后面介绍的 TCP 数据段中的“窗口大小”字段），接收端利用该字段向发送端通报当前还可接收的 TPDU 数量，即接收缓冲区大小，发送端则按当前接收缓冲区容量发送适量的 TPDU。

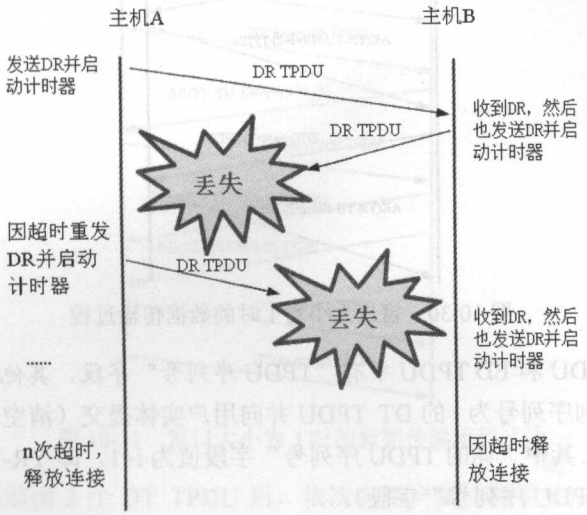


图 10-29 另一方 DR TPDU 和重发 DR TPDU 丢失情况下的非正常连接释放流程

在建立连接时，双方通过 CR TPDU 和 CC TPDU 中的 CDT 字段捎带着相互通告各自的初始窗口大小，即初始信用量。在数据传输过程中，发送端按接收端发回的 AK TPDU 中的 CDT 值发送一定的数据量；接收端可以根据接收缓冲区的使用状况动态地调整接收窗口大小，并在发送 AK TPDU 进行确认时捎带着将新的窗口大小通告给发送端。发送端将按新的接收窗口尺寸来调整发送窗口大小，接收端也用新的接收窗口大小来验证新到达数据分组的可接受性。下面举两个例子来说明一下。为了简便起见，假设双方的初始窗口大小相同。

1. 窗口大小为 1 时的数据传输过程

当在传输连接建立时，发送端发送的 CR TPDU 和接收端发送的 CC TPDU 中的 CDT 值均为 1（0001）时，表明发送端和接收端的窗口大小为 1，即发送端发送一个 TPDU 后必须等到接受了该 TPDU 的应答 TPDU 后才能继续传送新的 TPDU；接收端也类似，即仅当发送端发送一个 TPDU 后接收端才能向发送端返回一个确认 TPDU。也就是发送端发送一个，接收端必须确认一个，如图 10-30 所示。其实这与数据链路层的“停－等”窗口协议原理是一样的。

现假设发送端主机 A 和主机 B 的初始 TPDU 序列号分别为 i 和 j，具体步骤描述如下：

（1）主机 A 向主机 B 发送第一个 DT TPDU，序列号为 i（seq=i）。因为窗口大小为 1，所以必须等待对方收到并发回确认后才能继续发送下一个 DT TPDU。

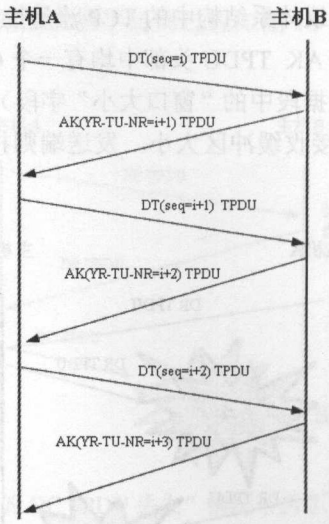


图 10-30　窗口大小为 1 时的数据传输过程

【注意】仅 DT TPDU 和 ED TPDU 中有“TPDU 序列号”字段，其他 TPDU 均没有。

（2）主机 B 在收到序列号为 i 的 DT TPDU 并向用户实体提交（清空缓存）后向主机 A 发送第一个确认 AK TPDU，其中“你的 TPDU 序列号”字段值为 i+1，即 YR-TU-NR=i+1（AK TPDU 和 EA TPDU 等均无“TPDU 序列号”字段）。

（3）因为主机 B 已把所收到的 DT TPDU 向用户实体提交了，缓存为空，可继续向主机 B 发送一个 DT TPDU，序列号为 i+1（seq=i+1）。此时同样因窗口大小为 1，不能继续发送，需要等待主机机 B 的确认 TPDU。

（4）主机 B 在收到序列号为 i+1 的 DT TPDU 并向用户实体提交（清空缓存）后向主机 A 发送第一个确认 AK TPDU，其中“你的 TPDU 序列号”字段值为 i+2，即 YR-TU-NR=i+2。

以后的步骤就与前面的一样了，主机 A 发送一个 DT TPDU 后就停下来，只有在收到主机 B 的确认 TPDU 后才继续发送下一个 DT TPDU。这种情况下的特点是：只有发送端收到一个确认 TPDU 才可以再发送一个新的 DT TPDU。

当然这种情况比较少见，因为这种窗口机制不能有效地利用信道资源。

2. 窗口大小为 3 时的数据传输过程

适当调整窗口大小则可以既使接收端和发送端速率相匹配，又可以充分利用现有带宽资源，提高传输效率。现假设通过在 CR TPDU 和 CC TPDU 中设置 CDT 为 3，即双方的窗口大小均为 3，则发送端可以在第一次连续发送 3 个 DT TPDU 后再停下来等待接收端的一个或多个确认信息，接收端也类似。但是发送端和接收端并不是每次都可以连接发送 3 个 TPDU，要视对方返回的确认 TPDU 确认数多少而定，其实也就是要视对方当前可用缓存空间大小而定。

如图 10-31 所示是在窗口大小为 3 的情况下的数据传输流程，具体过程描述如下：

（1）主机 A 向主机 B 连续发送 3 个 DT TPDU，序列号分别为 i（seq=i）、i+1（seq=i+1）和

i+2（seq=i+2）。因为窗口大小为 3，所以必须等待对方收到并发回确认（相当于清空部分或全部缓存）后才能继续发送下一个 DT TPDU。

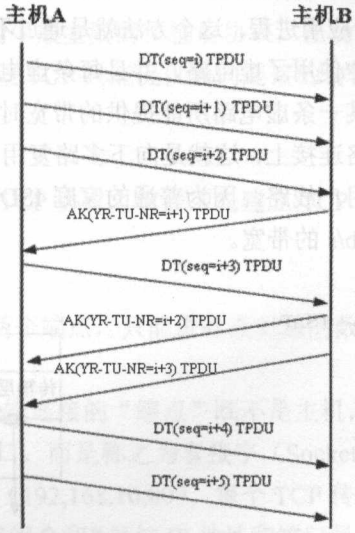


图 10-31　窗口大小为 3 时的数据传输流程

（2）主机 B 在收到前面 3 个 DT TPDU 后，依次向用户实体提交并依次向主机 A 发送确认 AK TPDU。现假设主机 A 收到了主机 B 的第一个 AK TPDU，“你的 TPDU 序列号”字段值为 i+1。此时在 AK TPDU 中 CDT 为 1，表示可以再接收一个 TPDU。

（3）在收到主机 B 发来的第一个 AK TPDU 后，看到其中的 CDT 值为 1，于是马上向主机 B 发送下一个 DT TPDU，序列号为 i+3（seq=i+3）。此时因为还没有收到新的来自主机 B 的 AK TPDU，表明对方没有可用缓存空间了，需要停下来，继续等待来自主机 B 的新的确认 TPDU。

（4）现假设主机 A 在很短的间隔时间内几乎同时收到了两个 AK TPDU，“你的 TPDU 序列号”字段值分别为 i+2 和 i+3。主机 A 从后面收到的 AK TPDU 中的 CDT 值得知主机 B 目前可以再接收两个 TPDU，于是又连接向主机 B 发送两个 DT TPDU，序列号分别为 i+4（seq=i+4）、i+5（seq=i+5）。

随后的步骤与前面类似了，总之一个原则是，发送端的主机 A 收到了来自接收端的主机 B 的多少个确认 TPDU，则可继续发送多少个 DTP TPDU，发送窗口大小不是固定的。

传输层还具有数据缓冲的功能，虽然这也与数据链路层的数据缓冲功能类似，但所采取的措施不一样。传输层的缓存策略和管理方法包括：缓存方式和策略、发送端申请/接收端分配、接收端在应答中夹带新的分配信息、阻塞和潜在死锁等几个方面，在此不作具体介绍了。

10.2.7　多路复用

在传输层也可以采取多路复用（Session Multiplexing）技术来提高传输连接的利用率。传输层的多路复用可以是由多个传输连接复用同一个网络连接的向上多路复用方式，也可以是由一个传输

连接在多个网络连接上循环传输的向下多路复用方式。

假设一台机器只一个网络地址可以使用，那么这台机器上的所有传输连接都必须使用这一个网络地址，这就是向上多路复用方式，如图 10-32 所示。当一个 TPDU 进来的时候，传输实体需要用一种方法来指明应该将它交给哪个应用进程，这个方法就是通过不同传输连接所使用的不同端口号。

还有一种情况，在一个子网中使用了虚电路，并且每条虚电路上有最大的数据传输速率限制，当某个传输连接所需的带宽超过某一条虚电路所能提供的带宽时，就需要把这一个传输连接以轮循的方式把数据流分配在不同的网络连接上，这就是向下多路复用方式，如图 10-33 所示。向下多路复用方式的典型应用就是 ISDN 用户线路，因为普通的家庭 ISDN 中每条线路只有 64Kb/s 带宽，通过复用两条线路最终达到 128Kb/s 的带宽。

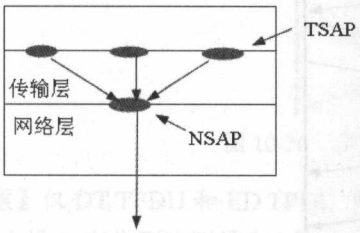


图 10-32　向上多路复用示意图

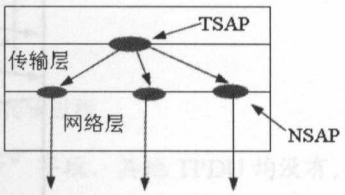


图 10-33　向下多路复用示意图

10.2.8　崩溃恢复

在传输连接中遇到由于主机或路由器不能正常工作，造成 TPDU 传输过程中断的现象就是崩溃。崩溃恢复就是要恢复到崩溃前的工作状态，继续 TPDU 传输。要实现这样一项功能就必须解决在崩溃前发送的最后一个 TPDU 是否需要重传的问题。

如果网络层提供数据报服务，则需要传输实体留有丢失的 TPDU 的副本；如果网络层提供面向连接的服务，则重建新的虚电路，重发丢失的 TPDU。要从主机或服务器崩溃中恢复，存在着复杂的问题，在此不作讨论。

总体来说，从第 N 层崩溃中恢复只能由第 N+1 层来完成，并且只能在第 N+1 层留有足够状态信息的情况下才能完成。

对于传输层崩溃恢复的方法需要了解在崩溃前发送主机和接收主机的状态，以及发送主机对最后 TPDU 的处理。崩溃恢复工作必须由更高层次（应用层）来完成，具体的连接恢复机制参见 10.4 节将要介绍的 TCP 连接可靠性。

10.3　TCP 协议基础

前面主要介绍了 OSI/RM 体系结构中的传输层服务，本节要具体介绍 TCP/IP 协议体系结构中的主要传输层协议——TCP（传输控制协议），其中包括 TCP 协议的一些主要特性和在本章前面介

绍的各种传输服务功能实现原理。

10.3.1 TCP 主要特性

TCP 是 TCP/IP 协议体系结构中最主要的传输层协议，在功能上与本章前面介绍的 OSI/RM 网络中的 TP4 类传输层协议类似，但 TCP 只能提供面向连接的传输服务，而且 TCP 还具有以下主要特性：

（1）面向连接的传输协议。

也就是说，应用程序在使用 TCP 协议之前必须先建立 TCP 传输连接，在传输数据完毕后又必须释放已建立的 TCP 传输连接。

（2）仅支持单播传输。

每条 TCP 传输连接只能有两个端点，只能进行点对点的数据传输，不支持多播（Multicast）和广播（Broadcast）传输方式。

【说明】这里所说的 TCP 传输连接的“端点”既不是主机，也不是主机的 IP 地址，也不是应用进程，同样不是传输层协议端口，而是称之为套接字（Socket）。套接字是 IP 地址和端口号的组合，中间用冒号或逗号分隔，如（192,168.10,80）。每个 TCP 传输连接有两个端点，也就是有两个套接字，即“源 IP 地址和端口号”组合和“目的 IP 地址和端口号”组合，可表示为{socket1,socket2}或者 {(IP1,Port1)，(IP2,Port2)}。

（3）提供可靠的交付服务。

也就是说通过 TCP 连接传送的数据可以无差错、不丢失、不重复，且按时序到达对端。

（4）传输单位为数据段。

TCP 仍采用了传统的“数据段”作为数据传输单元。数据段大小由应用层传送的报文大小和所途经网络中 MTU 值的大小决定，所以每次发送的 TCP 数据段大小是不固定的。在一个具体的网络中，有一个 MSS（Maximum Segment Size，最大数据段大小），最小的数据段可能仅有 21 个字节（其中 20 个字节属 TCP 数据段头部，数据部分仅 1 个字节）。

（5）仅一种 TPDU 格式。

虽然 TCP 的数据传输单元称为“数据段”，其实也可以按照 OSI/RM 的说法称为 TPDU，但在 TCP 协议中只有一种 TPDU，而不会像 OSI/RM 中有 10 多种不同格式的 TPDU（具体参见 10.1.5 节至 10.1.7 节），因为在 TCP 数据段头部已包括了各种 TPDU 所需的特征字段，主要是通过其中的多个控制位来实现的，具体将在下一节介绍 TCP 数据段的格式时介绍。

（6）支持全双工传输。

TCP 允许通信双方的应用程序在任何时候都能发送数据，因为 TCP 连接的两端都设有发送和缓存，用来临时存放双向通信的数据。当然 TCP 可以立即发送一个数据段，也可以缓存一段时间以便在一次发送更多的数据段（最大的数据段大小是由 MSS 决定的）。

（7）TCP 连接是基于字节流而非报文流的。

TCP 不是像本章后面将要介绍的 UDP 协议那样以一个个报文方式独立地进行传输，而是在不

保留报文边界的情况下以字节流方式进行传输。

（8）每次发送的 TCP 数据段大小和数据段数都是可变的。

因为在 TCP 协议中，每次发送多少字节的数据不是固定的，也不是由主机当前可用缓存决定的，而是根据对方给出的窗口大小和当前网络的拥塞程度来决定的（本章后面将要介绍的 UDP 协议所发送的报文长度是由应用进程给出的），所以每次可以发送的 TCP 数据大小是不固定的。

TCP 的传输单元是数据段，数据段又是在应用层传送的数据基础上进行封装的，所以每次的 TCP 数据段大小是由不同的应用进程数据决定的。当然如果应用层传送的报文太长，超出了 MSS，则需要对报文进行分段，所以总的来说，每次传输的 TCP 数据段大小是由应用层数据和 MSS 双重决定的。

这样一来，每次可以发送的 TCP 数据段数也是不固定的，可以一次仅发送一个 TCP 数据段，可以一次发送多个 TCP 数据段，只要在当前可用的“发送窗口大小”限制之内即可。另外，如果应用进程传送到 TCP 缓存中的数据太长，TCP 可以对它进行分段，反之如果传到 TCP 缓存中的数据太小，则 TCP 会等待缓存中有足够多的数据后再组装成一个数据段一起发送。

10.3.2 TCP 数据段格式

前面说了，TCP 的协议数据单元仍采用传统意义上的 Segment（分段）这种叫法，而没有使用 OSI/RM 中的 TPDU 叫法，当然两者本质上其实是一致的。所使用的 TCP 软件决定了数据段的大小，可以将多次写操作中的数据累积起来放到一个数据段中，也可以将一次写操作的数据分割成多个数据段。有两个因素决定了数据段的大小：一是每个 TCP 数据段的大小必须符合 IP 数据包的 65515 个字节的有效载荷大小限制要求（一个 IP 数据报最大不能超过 65535 个字节，即 64KB，一个 IP 协议报头最小是 20 个字节）；二是每个网络都有一个 MTU 值（其实是对应下面具体数据链路层协议帧“数据”字段的大小），所以每个 TCP 数据段必须适合于 MTU 的限制要求。

TCP 通过数据段的交互来建立连接、传输数据、发出确认、进行差错控制和流量控制及关闭连接。整个 TCP 数据段也分为“数据段头”和“数据”这两部分（这也是绝大多数报文封装的方式），所谓“数据段头”就是 TCP 为了实现端到端可靠传输所加上的 TCP 协议控制信息，而“数据”部分则是指由高层（即 TCP/IP 协议体系结构中的“应用层”）来的用户数据。但 TCP 中只有一种 TPDU 格式，所有类型的数据段格式都统一在如图 10-34 所示的 TCP 数据段格式中，不同类型的数据段是通过其中的多个控制位来实现的。下面是其中各字段的具体说明。

（1）源端口（Source Port）和目的端口（Destination Port）。

分别代表呼叫方和被叫方的 TCP 端口号，各占 16 位。一个端口与其主机的 IP 地址就可以完整地标识一个端点了，也就是构成了前面所说的套接字（Socket）。

（2）序号（Sequence Number）。

指 TCP 数据段中的“数据”部分（不包含“数据段头”部分）的第一个字节的序号，占 32 位。在一个 TCP 连接中，传送的数据字节流中的每一个数据字节都要按顺序进行编号，整个要传送的字节流的起始序号必须在连接建立时设置。例如一个数据段的“序号”字段值是 101，而该数字段

中共有 100 个字节，则表明本数据段的最后一个字节的编号是 200。这样一来，下一个数据段的“序号”字段值就应该是 201 而不是 102。这一点需要特别注意。

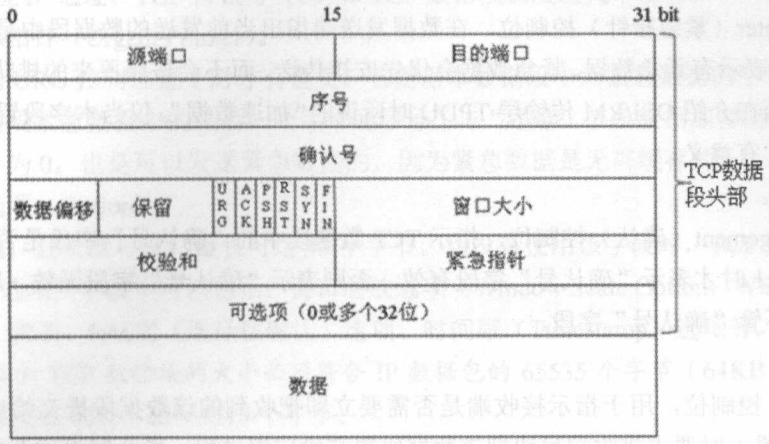


图 10-34 TCP 协议数据段格式

（3）确认号（Acknowlegment Number）。

指期望接收到对方下一个数据段中“数据”部分的第一个字节序号，占 32 位。注意，“确认号”不是代表已经正确接收到的最后一个字节的序号，仅表示在此数值之前的所有数据对方已连续正确接收了，因为在两个已正确接收的数据段之间可能会有一个或多个数据还没有正确接收。

例如，主机 A 向主机 B 发送的一个数据段，其“序号”为 101，而该数据段“数据”部分的长度假设是 100 个字节。当主机 B 正确接收后可能会立即向主机 A 返回一个 ACK 确认数据段。这时在 ACK 数据段中的“确认号”字段值是多大呢？答案是 201，而不是 102。

再假设，主机 B 在正确接收到序号为 101、201 这两个数据段后还正确接收到序号为 501、601 这两个数据段，那么主机 B 如果现在想对主机 A 发送 ACK 数据段进行确认的话，ACK 数据段中的“确认号”是多少呢？答案是 301，尽管后面的 401、601 数据段都正确接收了，但中间的 301、401 这两个数据段没有正确接收到。

以上“序号”和“确认号”两个字段共同用于 TCP 服务中的差错控制，确保 TCP 数据传输的可靠性，具体将在本章后面介绍。

（4）数据偏移。

TCP 数据段中的“数据”字段起始处距离整个 TCP 数据段起始处有 4 个字节（即是以 4 个字节为单位进行计算的）的偏移量，用于确定 TCP 数据段头部的长度（有些抓包软件称之为“头部长度”字段，如 Wireshark），占 4 位。因为 TCP 数据段头中有不确定的“可选项”字段，所以“数据偏移”字段是非常必要的，到了接收方要向上层传输数据时就知道所去掉的 TCP 协议头有多大。因为 4 个比特位可以表示的最大数为 15，所以数据偏移量最大为 60 个字节，这也是 TCP 数据段头部的最大长度。

（5）保留（Reserved）。

这是为将来应用而保留的 6 个比特位，目前应全设置为 0。

（6）URG。

Urgent Pointer（紧急指针）控制位，在数据发送端指出当前发送的数据段中是否有紧急数据，占 1 位，置 1 时表示有紧急数据。紧急数据会优先安排传送，而不会按照原来的排队顺序进行发送。相当于本章前面在介绍 OSI/RM 传输层 TPDU 时所说的“加速数据”。仅当本字段置 1 时后面的“紧急指针”字段才有意义。

（7）ACK。

Acknowledgement（确认）控制位，指示 TCP 数据段中的“确认号”字段是否有效，占 1 位。仅当 ACK 位置 1 时才表示“确认号”字段有效，否则表示“确认号”字段无效，应用层实体在读取数据时可以不管“确认号”字段。

（8）PSH。

Push（推）控制位，用于指示接收端是否需要立即把收到的该数据段提交给应用进程，占 1 位。当 PSH 位置 1 时要求接收端尽快把该数据段提交给应用进程，置 0 时没这个要求，可以先缓存起来。

（9）RST。

Reset（重置）控制位，用于重置、释放一个已经混乱的传输连接，然后重建新的传输连接，占 1 位。当 RST 位置 1 时释放当前传输连接，然后可以重新建立新的传输连接。

（10）SYN。

Synchronization（同步）控制位，用来在传输连接建立时同步传输连接序号，占 1 位。当 SYN 位置 1 时，表示这是一个连接请求或连接确认报文。当 SYN=1 而 ACK=0 时，表明这是一个连接请求数据段（相当于本章前面介绍的 OSI/RM TPDU 中的 CR TPDU）。如果对方同意建立连接，则对方会返回一个 SYN=1，ACK=1 的确认（相当于本章前面介绍的 OSI/RM TPDU 中的 CC TPDU）。

（11）FIN。

Final（最后）控制位，用于释放一个传输连接，占 1 位。当 FIN 位置 1 时，表示数据已全部传输完成（可以是发送过的，也可以是接收完）。当发送端没有数据要传输时可要求释放当前连接，但接收端仍然可以继续接收还没有接收完的数据。正常传输时该位置 0。

（12）窗口大小。

指示发送此 TCP 数据段的主机上用来存储传入数据段的窗口大小，亦即发送者当前还可以接收的最大字节数，相当于本章前面介绍的 OSI/RM TPDU 中的 CDT（信用量分配），占 16 位。

TCP 的“窗口大小”字段与本章前面介绍的 CDT 一样是使用可变大小的滑动窗口协议来进行流量控制。“窗口大小”字段的值告诉接收本数据段的主机，从本数据段中所设置的“确认号”值算起，本端目前允许对端发送的字节数，是作为让对方设置其发送窗口大小的依据。假设本次所发送的数据段的“确认号”字段值为 501，而在数据段的“窗口大小”字段值是 100，则从 501 算起，本端还可以接收 100 个字节（字节序号是 501～600）。

（13）校验和（Checksum）。

对“数据段头”“数据”和“伪头部”这三部分进行校验，占 16 位。“伪头部”包括源主机和目的主机的 32 位 IP 地址、TCP 协议号（6）和 TCP 数据段长度这几个部分。

（14）紧急指针（Urgent Pointer）。

仅当前面的 URG 控制位置 1 时才有意义，它指出本数据段中为紧急数据的字节数，占 16 位。它指出了本数据段紧急数据中的最后一个字节的序号（与“序号”字段值相加得出），占 16 位。即使当前窗口大小为 0，也是可以发送紧急数据的，因为紧急数据是无需缓存的。

（15）可选项（Option）。

该字段可选，且长度可变，最长可达 40 个字节。当没有使用该字段时，TCP 头部的长度就是 20 个字节。“可选项”字段中可以包括：窗口缩放选项（Window Scale Option，WSopt）、MSS（最大数据段大小）选项、SACK（选择性确认）选项、时间戳（Timestamp）选项等。

【说明】每个 TCP 数据段的大小必须符合 IP 数据包的 65535 个字节（64KB）的有效载荷大小限制要求，最大为 65535-20=65515 个字节。

（16）数据（Data）。

这是由应用层应用进程提交的数据，作为 TCP 数据段的“数据”（有效载荷）部分。

10.3.3 TCP 套接字

在本章前面介绍 OSI/RM 传输层服务时，我们说到传输通信两端的端点就是 TSAP 地址，对应的就是传输层协议端口。在 TCP/IP 网络中同样有端点的概念，但它并不是直接采用 OSI/RM 中所说的 TSAP 叫法，而是称为“套接字”（Socket），就像在 TCP 协议中仍然把所传输的数据称为“数据段”，而没有采用 OSI/RM 中的叫法 TPDU 一样。当然，需要说明的是，Socket 并不能直接等同于 TSAP，只是类似，实际上 Socket 只是包括了 TSAP 地址——传输层端口。“套接字”最早使用于 UNIX 操作系统中，后来被广泛地应用于 Windows 和 Linux 系统中，成了事实上的 TCP 协议标准。

1. 套接字概述

在 TCP/IP 网络中，区分不同应用程序进程间的网络通信和连接主要有 3 个参数：通信的目的 IP 地址、使用的传输层协议（TCP 或 UDP）和使用的端口号（此处说明，Socket 不仅在 TCP 协议中有，在 UDP 协议中同样有）。通过将这 3 个参数结合起来与一个 Socket 进行绑定，应用层就可以和传输层通过套接字接口来区分来自不同应用程序进程或网络连接的通信，实现数据传输的并发服务。为了区别不同的应用程序进程和连接，许多计算机操作系统为应用程序与 TCP/IP 协议交互提供了称为套接字（Socket）的接口

Socket 可以看成在两个网络应用程序间进行通信连接的一个端点（或者说是“逻辑接口”），是连接应用程序和网络驱动程序的桥梁，如图 10-35 所示。从位置上看，它与本章前面介绍的 TSAP 是不一样的，因为 TSAP 位于传输层上边缘（但仍属于传输层），而 Socket 完全位于应用层，但它调用了传输层的端口。Socket 包括了 TSAP 地址，同样它还包括了在本章前面所介绍的服务原语，

具体将在本节后面介绍。

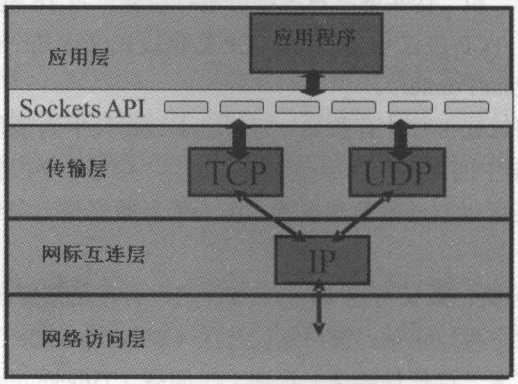


图 10-35　Socket 接口的位置

在 TCP/IP 通信中，应用程序把数据传送给 Socket，然后由 Socket 通过传输层向下提交给网络驱动程序向网络上发送出去。计算机从网络上收到与该 Socket 绑定 IP 地址和端口号相关的数据后，由网络驱动程序通过传输层向上提交给应用层的 Socket，最后应用程序从 Socket 中提取所要接收的数据。通信双方计算机上的网络应用程序就是这样通过 Socket 进行数据的发送与接收的，基本交互过程如图 10-36 所示。

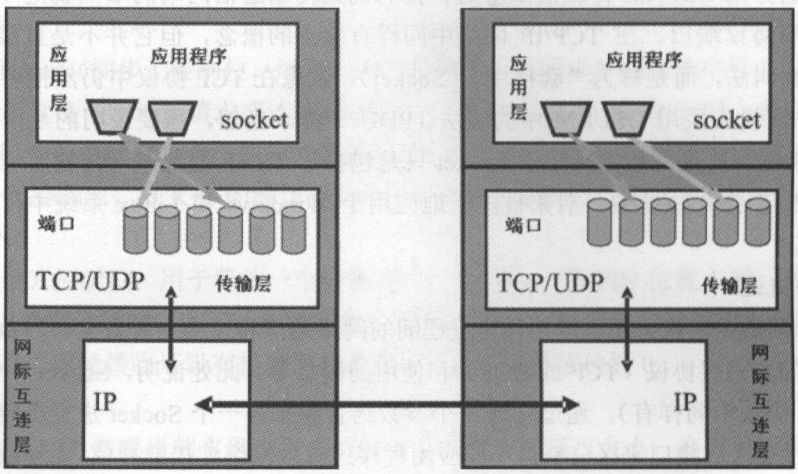


图 10-36　Socket 在 TCP/IP 通信中的交互应用

从图中可以看出，应用层上针对每个应用进程都有一个 Socket，调用传输层中的一个特定端口（当然，这个端口对于常规服务来说是固定的，对于非常规服务来说不是固定的）。也就是应用层的 Socket 和传输层的端口都非常多，但位于网络层，对于一台具体的主机和具体的网络来说，它的 IP 地址却是唯一的，也就是说 Socket 和端口与 IP 地址之间是“多对一”的关系。

2. Socket 原语

在 TCP Socket 中包括了如表 10-6 所示的一组（共 8 个）TCP 数据传输服务原语。表中前 4 个原语在服务器端要按顺序执行（但并不是说服务器仅调用这 4 个原语）。SOCKET 原语用来创建一个新的端点，并且在传输实体中为它们分配相应的空间。SOCKET 原语调用成功后返回一个普通的文件描述符，以便在后续其他的原语调用中使用。但是新创建的套接字并没有网络地址，需要通过 BIND 原语为它进行分配，所以 BIND 原语的用途就是为新建的套接字绑定一个本地网络地址。

表 10-6 TCP Socket 原语

原语名称	含义
SOCKET	创建一个新的通信端点
BIND	将一个本地地址关联到一个套接字上
LISTEN	通告愿意接受连接并给出队列大小
ACCEPT	阻塞调用方，直到有人企图连接上来
CONNET	主动尝试建立一个连接
SEND	在指定的连接上发送数据
RECV	从指定的连接上接收数据
CLOSE	释放指定的连接

BIND 原语执行后就需要对 LISTEN 原语进行调用，它为新建的套接字分配一定的缓存空间，以便为后面要进来的连接进行排队，这样就可以使多个客户可以同时对一个服务器进行访问。但此时服务器还是不是阻塞状态，也就是服务器仍没有进入等待连接状态，这里需要再调用一个 ACCEPT 原语。当一个请求连接的 TCP 数据段到来的时候，传输实体会通过调用 ACCEPT 原语创建一个新的套接字并返回一个与其关联的文件描述符。这个新的套接字与原来由 SOCKET 原语创建的套接字具有相同的属性。这时，服务器就可以调用一个进程来处理这个套接字上新来的连接，而服务器本身又回去继续等待原套接字上的下一个连接。

以上介绍的是由服务器调用的 4 个原语，在客户端使用的原语主要包括：SOCKET、CONNECT、SEND、RECV 这 4 个。首先也是先由 SOCKET 原语创建一个新的套接字，但由于服务器并不关心它所用的网络地址，所以客户端不必调用 BIND 原语。然后通过 CONNECT 原语阻塞客户端，并主动发起一个 TCP 传输连接。当 CONNECT 原语调用完成后，即收到服务器端发来的确认数据段后，客户端进入被解除阻塞状态，建立传输连接。现在双方都可以使用 SEND 和 RECV 原语在建立起来的全双工连接上进行数据的收与发。

表 10-6 中最后的 CLOSE 原语是服务器和客户端都可以调用的，用来释放当前 TCP 传输连接。当双方都执行了 CLOSE 原语后，连接才会真正释放。

以上服务器与客户端的原语基本调用流程如图 10-37 所示。

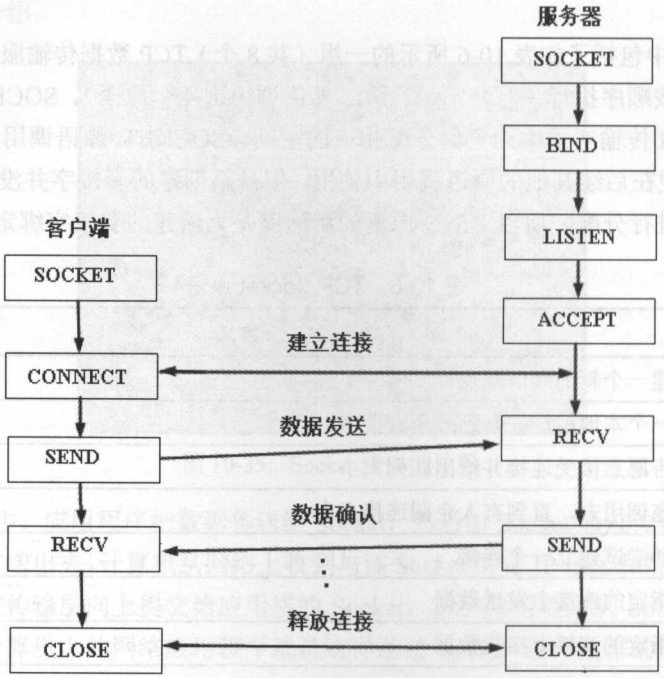


图 10-37　TCP Socket 原语调用基本流程

10.3.4　TCP 端口

我们在本章前面已经说到，传输层 TSAP 地址就是传输层协议端口，是对网络应用进程的一种标识。在这里要区分“端口”与“接口”两个不同的概念，接口是物理的，而“端口”却是一种抽象的软件结构，包括一些数据结构和 I/O（输入/输出）缓冲区，故属于软件接口范畴，是传输层 TSAP 地址。

每个端口都拥有一个称为“端口号”的整数描述符，用来标识不同的端口或进程。在 TCP 数据段格式中定义了一个 16 比特长度的“源端口”和“目的端口”两个字段，也就是说可以定义 2^{16} 个端口，其端口号从 0 到 2^{16}-1（65535）。由于 TCP/IP 传输层的 TCP 和 UDP 两个协议是两个完全独立的软件模块，因此各自的端口号也相互独立，即各自可独立拥有 65535 个端口。每种应用层服务或应用进程都具有与传输层唯一连接的端口，并且使用唯一的端口号进行区分。

在整个 65535 个 TCP 端口中主要分为以下三大类（是由 IANA 管理的）：

（1）保留端口。

通常是把 0～1023 号 TCP 端口用于保留，这类端口也称为“常规端口”或者“公认端口”（Well-known Port）。这些端口基本上都已固定地被分配给了已知的网络应用协议，如 HTTP 协议（对应 WWW 服务）中的 80 号端口、Telnet 协议中的 23 号端口、SMTP 协议中的 25 号端口等。目前，这一类端口的端口号分配已经被广大网络应用者所接受，形成了标准，在各种网络的应用中

调用这些端口号就意味着使用它们所代表的应用协议，不能被分配给其他网络应用协议使用。表 10-7 所示为部分常规服务 TCP 端口及所对应的服务。

表 10-7 部分常规服务 TCP 端口

端口号	对应的协议	用途说明
20	FTP	FTP 数据传输
21	FTP	FTP 控制消息
22	SSH	安全登录
23	Telnet	远程登录
25	SMTP	邮件发送
42	WINS	Internet 名称解析服务
53	DNS	域名解析
69	TFTP	小型文件传输协议
79	Finger	查询有关一个用户的信息
80	HTTP（或 WWW）	万维网服务
110	POP3	邮件接收服务
115	SFTP	简单文件传输协议
119	NNTP	UESNET 新闻
137	NetBIOS	NetBIOS 名称服务
138	NetBIOS	NetBIOS 数据报服务
139	NetBIOS	NetBIOS 会话服务
143	IMAP	邮件消息管理
500	ISAKMP	Internet 安全关联和密钥管理
520	RIP	基于距离矢量的动态路由

（2）动态分配端口。

这种动态端口的端口号一般都大于 1024，没有被分配给固定的网络应用服务，可以被动态地分配给任意网络服务应用程序使用。也就是说，在使用网络应用软件访问网络的时候，网络应用软件可以向系统申请一个大于 1024 的端口临时代表这个软件与传输层交换数据，并且使用这个临时的端口与网络上的其他主机通信。

动态分配的端口中有许多是被木马类黑客程序所利用的，通过查看某个端口是否被占用就可以了解当前系统是否中了某一木马。

（3）注册端口。

注册端口比较特殊，它也是固定为某个应用服务的端口，但是它所代表的不是已经形成标准的

应用层协议，而是某个软件厂商开发的应用程序，如 CCProxy 中的 8080 号端口。这类注册端口的端口号一般也都大于 1024。

当网络中的两台主机进行通信的时候，为了表明数据是由源端的哪一种应用发出的以及数据所要访问的是目的端的哪一种服务，TCP/IP 协议会在传输层封装数据段时把发出数据的应用程序的端口作为源端口，把接收数据的应用程序的端口作为目的端口，添加到数据段的头中，从而使主机能够同时维持多个会话的连接，使不同应用程序的数据不发生混淆。

10.3.5 TCP 连接的有限状态机

从表 10-6 中可以看出，TCP Socket 服务原语只有 8 个，比 OSI/RM 中定义的传输层服务原语还要少，但是在这 8 个 TCP Socket 服务原语中，有的原语又可以有不同的状态，如表 10-8 所示。我们把在 TCP 传输连接的建立和释放中的通信双方主机的这些状态称为“有限状态机”（Finite State Machine，FSM）。下面就来具体阐述一下在 TCP 传输连接的建立和释放过程中这些原语状态的变化。

表 10-8 TCP 连接状态

状态	描述
CLOSED	呈阻塞、关闭状态，表示主机当前没有活动的传输连接或正在进行传输的连接
LISTEN	呈监听状态，表示服务器正在等待新的传输连接进入
SYN RCVD	表示主机已收到一个传输连接请求，但尚未确认
SYN SENT	表示主机已经发出一个传输连接请求，等待对方确认
ESTABLISHED	传输连接建立，通信双方进入正常数据传输状态
FIN WAIT 1	（主动关闭）主机已经发送关闭连接请求，等待对方确认
FIN WAIT 2	（主动关闭）主机已经收到对方关闭传输连接确认，等待对方发送关闭传输连接请求
TIMED WAIT	完成双向传输连接关闭，等待所有分组消失
CLOSING	双方同时尝试关闭传输连接，等待对方确认
CLOSE WAIT	（被动关闭）收到对方发来的关闭传输连接请求并已确认
LAST ACK	（被动关闭）等待最后一个关闭传输连接确认并等待所有分组消失

图 10-38 描述了 TCP 通信主机在传输连接建立和释放过程中的各种有限状态机，图中方框中表示的是通信主机在不同时期的状态，箭头表示状态之间的转换，旁边的注释表示状态转换过程中所需进行的动作（包括调用 Socket 服务原语和 TCP 数据段的发送和接收等）。图中用粗线表示客户端主动与服务器端连接建立的正常过程，其中客户端的状态转移用带箭头的粗实线表示，服务器端的状态转换用带箭头的粗虚线表示，带箭头的细实线表示一些不常见的事件，如复位、同时打开、同时关闭等。

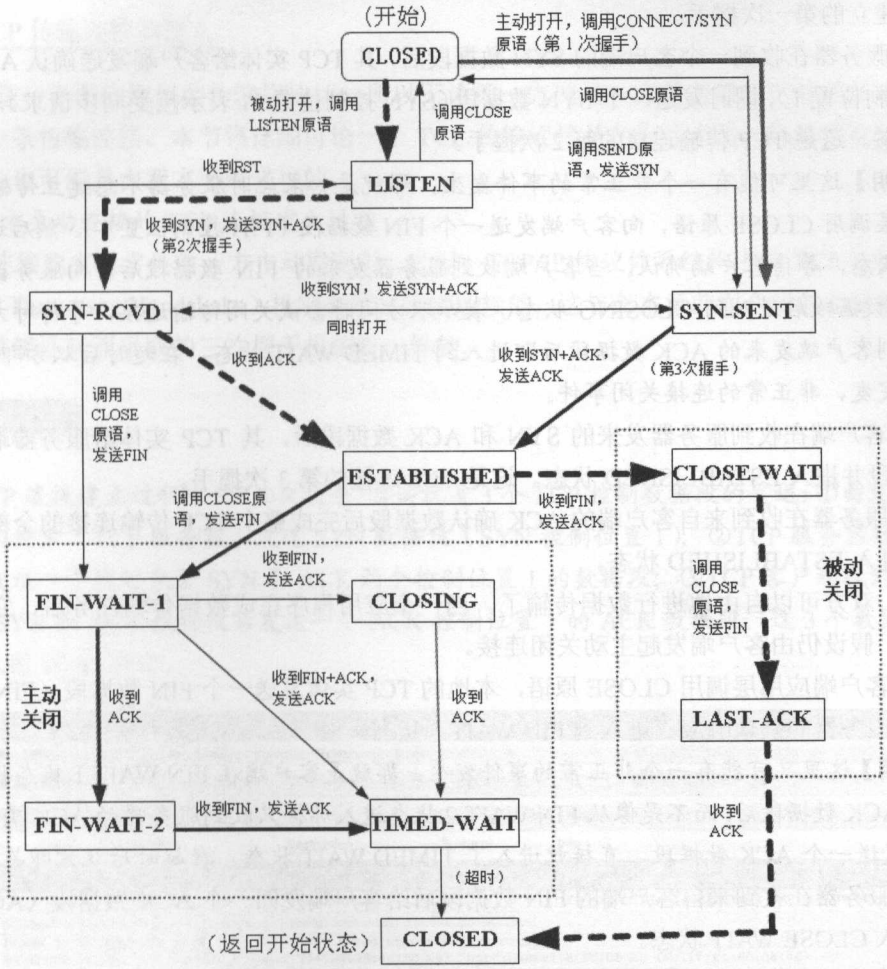


图 10-38 TCP 传输连接有限状态机转换流程

每个连接均开始于 CLOSED 状态。当一方执行了被动的连接原语（LISTEN）或主动的连接原语（CONNECT）时，它便会离开 CLOSED 状态。如果此时另一方执行了相对应的连接原语，连接便建立了，并且状态变为 ESTABLISHED。任何一方均可以先请求释放连接，当连接被释放后，状态又回到了 CLOSED。

为了看清楚客户端和服务器各自的状态转移流程，先沿着带箭头的粗实线路径来看客户端的状态转移过程，然后再沿着带箭头的虚实线路径来看服务器的状态转移过程。

（1）服务器应用层先调用 LISTEN 原语从 CLOSED 状态进入被动打开状态（LISTEN），等待客户端的连接。

（2）当客户端的一个应用程序调用 CONNECT 原语后，本地的 TCP 实体为其创建一个连接记录并标记为 SYN SENT 状态，然后给服务器发送一个 SYN 数据段（SYN 控制位置 1）。这是 TCP

传输连接建立的第一次握手。

（3）服务器在收到一个客户端的 SYN 数据段后，其 TCP 实体给客户端发送确认 ACK 数据段（ACK 控制位置 1），同时发送一个 SYN 数据段（SYN 控制位置 1，表示接受同步请求），进入 SYN RCVD 状态。这是 TCP 传输连接的第 2 次握手。

【说明】这里可能有一个非正常的事件发生，那就是如果此时服务器不想建立传输连接，则由其应用层调用 CLOSE 原语，向客户端发送一个 FIN 数据段（FIN 控制位置 1），然后进入到 FIN WAIT 1 状态，等待客户端确认。当客户端收到服务器发来的 FIN 数据段后，向服务器发送一个 ACK 确认数据段后进入到 CLOSING 状态，表示双方同时尝试关闭传输连接，等待对方确认。在服务器收到客户端发来的 ACK 数据段后即进入到 TIMED WAIT 状态，在超时后双方即关闭连接。这是一种突发、非正常的连接关闭事件。

（4）客户端在收到服务器发来的 SYN 和 ACK 数据段后，其 TCP 实体给服务器端发送一个 ACK 数据段并进入 ESTABLISHED 状态。这是 TCP 连接的第 3 次握手。

（5）服务器在收到来自客户端的 ACK 确认数据段后完成整个 TCP 传输连接的全部三次握手过程，也进入 ESTABLISHED 状态。

此时，双方可以自由地进行数据传输了。当一个应用程序完成数据传输任务后，它需要关闭 TCP 连接。假设仍由客户端发起主动关闭连接。

（6）客户端应用层调用 CLOSE 原语，本地的 TCP 实体发送一个 FIN 数据段（FIN 控制位置 1）并等待服务器的确认响应，进入到 FIN WAIT 1 状态。

【说明】这里又可能有一个非正常的事件发生，那就是客户端在 FIN WAIT 1 状态收到服务器的 FIN 和 ACK 数据段后（而不是像从 FIN WAIT 2 状态进入那样只收到服务器的 ACK 确认数据段）向服务器发送一个 ACK 数据段，直接就进入了 TIMED WAIT 状态。在超时后双方即关闭连接。

（7）服务器在收到来自客户端的 FIN 数据段后给客户端发回一个 ACK 数据段（ACK 控制位置 1），进入 CLOSE WAIT 状态。

（8）客户端在收到来自服务器的 ACK 确认数据段后就进入到了 FIN WAIT 2 状态，此时连接在一个方向上就断开了，但仍可以接收服务器端发来的数据段。

（9）当服务器收到客户端发来的 FIN 数据段时就知道客户端已有数据发送了，在本端已接收完全部的数据后，也由应用层调用 CLOSE 原语，请求关闭另一个方向的连接，其本地 TCP 实体向客户端发送一个 FIN 数据段并进入 LAST ACK 状态，等待最后一个 ACK 确认数据段。

（10）在客户端收到来自服务器的 FIN 数据段后，向服务器发送最后一个 ACK 确认数据段，进入 TIMED WAIT 状态。此时双方连接均已经断开，但 TCP 实体仍要等待一个 2 倍数据段 MSL（Maximum Segment Lifetime，最大数据段生存时间），以确保该连接的所有分组全部消失，防止出现确认丢失的情况。当定时器超时后，TCP 删除该连接记录，返回到初始状态（CLOSED）。

（11）服务器收到客户端最后一个 ACK 确认数据段后，其 TCP 实体便释放该连接并删除连接记录，也返回到初始状态（CLOSED）。

10.3.6 TCP 传输连接的建立

TCP 是一个面向连接的传输层协议，所以无论哪一方向另一方发送数据之前都必须先在双方之间建立一条传输连接。本节将详细讨论一个 TCP 传输连接是如何建立的，连接建立的发起可能是由一方，也可能是由双方同时进行的。

1. 单方主动连接的 TCP 连接建立过程

TCP 连接建立通常是由一方主动发起的。此时，TCP/IP 协议体系结构中的 TCP 协议也是使用三次握手（Three-way Handshake）机制来建立传输连接的，这与在本章前面介绍的 OSI/RM 传输层为了避免重复连接而采取的三次握手机制是一样的。



在 TCP 连接建立过程中的“三次握手”其实就是 3 个 TCP 控制数据段的发送：①由发起端（TCP 客户端）向对方（TCP 服务器）发送 SYN 数据段（SYN 控制位置 1）；②TCP 服务器接收到 SYN 数据段后发送一个同时包含 SYN 和 ACK 两个控制位置 1 的数据段；③TCP 客户端收到 TCP 服务器发来的 SYN 和 ACK 数据段后发送一个 ACK 控制位置 1 的 ACK 数据段。这 3 个数据段分别如图 10-39 至图 10-41 所示。

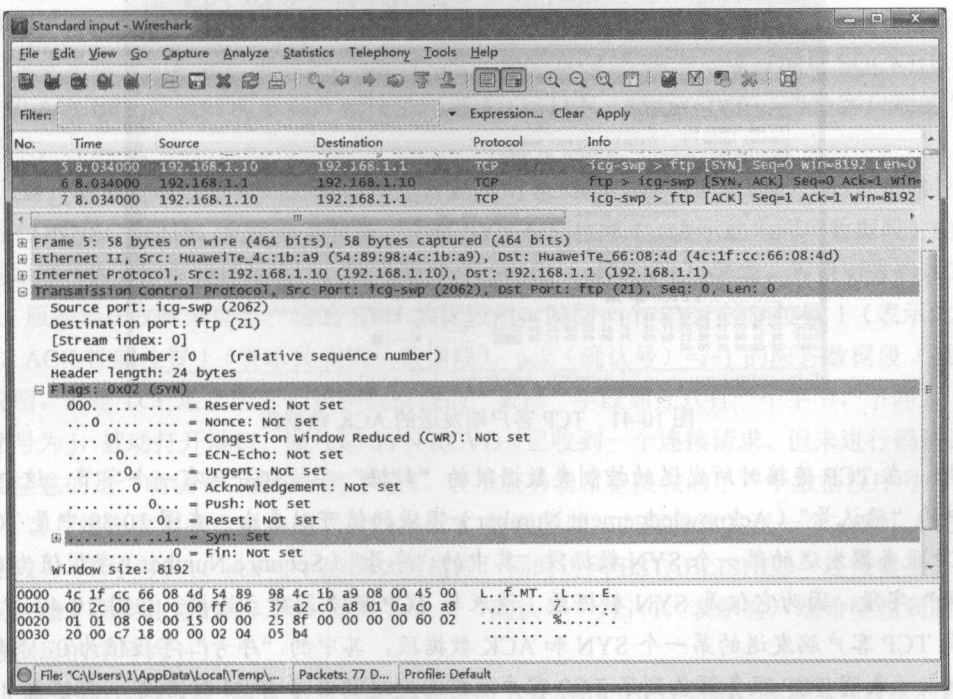


图 10-39 TCP 客户端第一次发送的 SYN 数据段

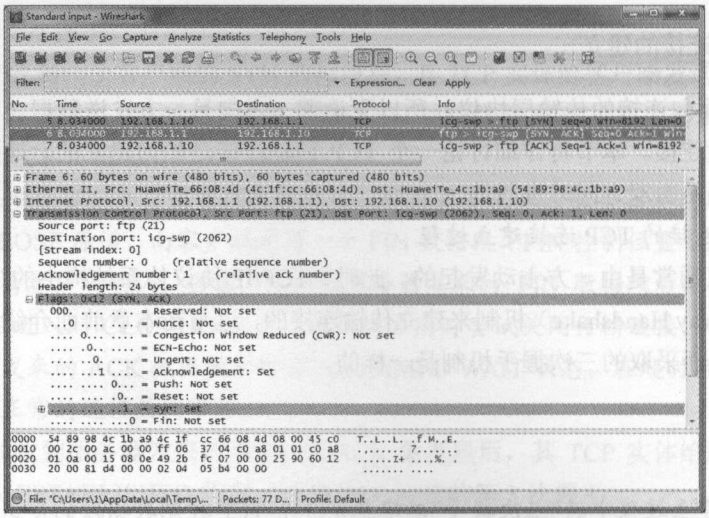


图 10-40　TCP 服务器端发送的 SYN 和 ACK 数据段

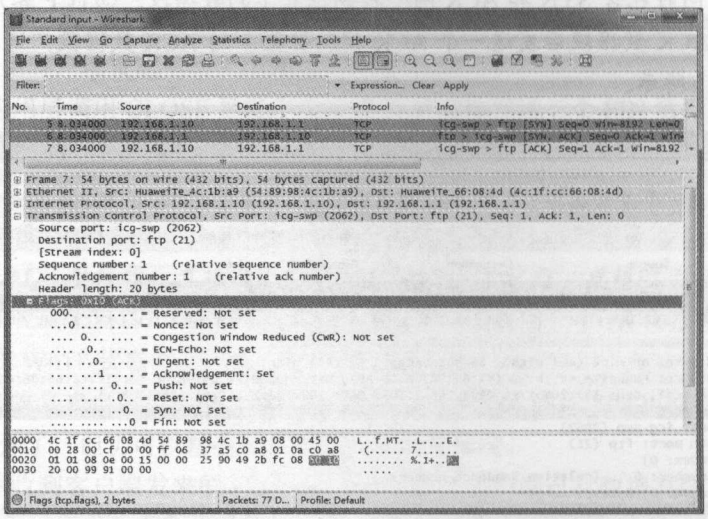


图 10-41　TCP 客户端发送的 ACK 数据段

另外，在 TCP 连接时所发送的控制类数据段的“数据”字段部分只有一个字节，这个从以上 3 个图中的“确认号”（Acknowledgement Number）字段的值可以看出。在图 10-39 中是 TCP 客户端向 TCP 服务器发送的第一个 SYN 数据段，其中的“序号”（Sequnce Number）字段值为 0，没有“确认号”字段，因为它仅是 SYN 数据段，请求与 TCP 服务器建立连接。在图 10-40 中是 TCP 服务器向 TCP 客户端发送的第一个 SYN 和 ACK 数据段，其中的“序号”字段值为 0，“确认号”字段值为 1（表明 TCP 服务器已到了 TCP 客户端前面发送的 0 号 SYN 数据段）；而在图 10-41 中是 TCP 客户端第二次向 TCP 服务器发送的 ACK 数据段，其中的“序号”字段值为 1，“确认号”

字段值为 1（表明 TCP 客户端已到了 TCP 服务器前面发送的 0 号 SYN、ACK 数据段）。但在真正的数据传输阶段，这时的“确认号”字段值要依据所接收的数据段的“数据”部分大小而定了。

下面是 TCP 连接建立的具体流程，如图 10-42 所示。

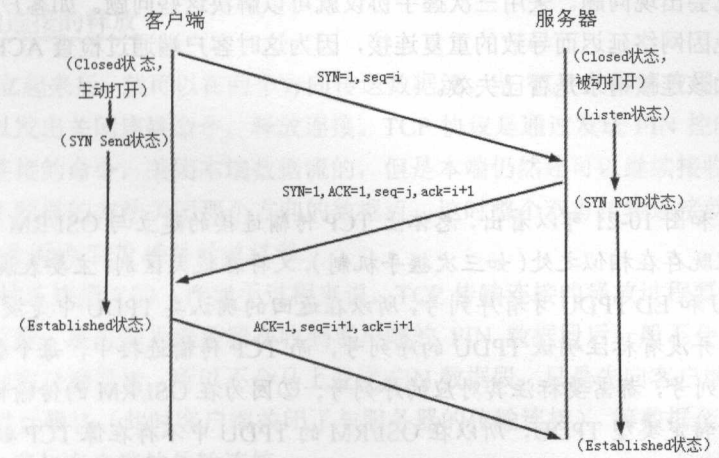


图 10-42　TCP 传输连接建立的三次握手过程

其实整体过程在图 10-38 中已有全面的体现，这里仅单独把 TCP 传输连接的建立过程列出来，具体步骤如下：

（1）首先是服务器初始化的过程，从 CLOSED（关闭）状态开始通过顺序调用 SOCKET、BIND、LISTEN 和 ACCEPT 原语创建 Socket 套接字，进入 LISTEN（监听）状态，等待客户端的 TCP 传输连接请求。

（2）客户端最开始也是从 CLOSED 状态开始调用 SOCKET 原语创建新的 Socket 套接字，然后需要调用 CONNECT 原语，向服务器发送一个将 SYN 控制位置 1（表示此为同步数据段）的数据段（假设初始序号为 i），主动打开端口，进入到 SYN SENT（已发送连接请求，等待对方确认）状态。

（3）服务器在收到来自客户端的 SYN 数据段后，发回一个 SYN 控制位置 1（表示此为同步数据段），ACK 控制位置 1（表示此为确认数据段），ack（确认号）=i+1 的应答数据段（原因在前面已有说明，就是 TCP 连接建立类控制数据段的“数据”字段部分只有一个字节，下同），同时假设初始序号为 j，被动打开端口，进入到 SYN RCVD（已收到一个连接请求，但未进行确认）状态。这里需要注意的是“确认号”是 i+1 而不是 i，表示服务器希望接收的下一个数据段序号为 i+1。

（4）客户端在收到来自服务器的 SYN+ACK 数据段后，向服务器发送一个 ack=1（表示此为确认数据段），序号为 i+1，ack=j+1 的确认数据段，同时进入 ESTABLISHED（连接建立）状态，建立单向连接。需要注意的是，此时序号为 i+1，“确认号”为 j+1，表示客户端希望收到服务器的下一个数据段的序号为 j+1。

（5）服务器在收到客户端的 ACK 数据段后进入 ESTABLISHED 状态，完成双向连接的建立。

连接可以由任一方或双方发起，一旦连接建立，数据就可以双向对等地流动，而没有所谓的主

从关系。三次握手是连接两端正确同步的充要条件，因为 TCP 建立在不可靠的分组传输服务之上，报文可能丢失、延迟、重复和乱序，因此协议必须使用超时和重传机制。如果重传的连接请求和原先的连接请求在连接正在建立时到达，或者当一个连接已经建立、使用和结束之后，某个延迟的连接请求才到达，就会出现问题。采用三次握手协议就可以解决这些问题。如客户端发送的 ACK 数据段就是为了避免因网络延迟而导致的重复连接，因为这时客户端通过检查 ACK 数据段中的“确认号”就可以得知该连接请求是否已失效。



对比图 10-42 和图 10-21 可以看出，总体上 TCP 传输连接的建立与 OSI/RM 中 TP 传输协议的传输连接建立过程既存在相似之处（如三次握手机制），又存在较大区别，主要表现在：①在 OSI/RM 中，只有 DT TPDU 和 ED TPDU 才有序列号，所以在返回的确认类 TPDU 中是没有 TPDU 序号的，所以在图 10-21 中并没有标注确认 TPDU 的序列号，而 TCP 传输连接中，每个数据段（无论是否携带数据）都有序列号，都需要标注其对应的序列号；②因为在 OSI/RM 的传输协议中不同服务原语可以使用不同的特定类型 TPDU，所以在 OSI/RM 的 TPDU 中不存在像 TCP 数据段中的 ACK、SYN 等之类的控制位，因为在 TCP 协议中所有数据段格式是一样的，不同的只是不同类型的 TCP 数据段的这些字段的取值不同；③TCP 传输连接中使用“确认号”字段与 OSI/RM TPDU 中的 YR-TU-NR（你的 TPDU 序列号）字段的功能是完全一样的，都是显示对端希望接收的下一个数据段的序列号，暗示该号码前面的所有数据段均已正确接收。

2. 双方同时主动连接的 TCP 连接建立过程

正常情况下，传输连接都是由一方主动发起的，但也有可能双方同时主动发起连接，此时就会发生连接碰撞，最终只有一个连接能够建立起来。因为所有连接都是由它们的端点进行标识的。如果第一个连接请求建立起一个由套接字(x,y)标识的连接，而第二个连接也建立了这样一个连接，那么在 TCP 实体内部只有一个套接字表项。

当出现同时发出连接请求时，两端几乎在同时发送一个 SYN 控制位置 1 的数据段并进入 SYN_SENT 状态。当每一端收到 SYN 数据段时，状态变为 SYN_RCVD，同时它们都再发送 SYN 控制位置 1，ACK 控制位置 1 的数据段，对收到的 SYN 数据段进行确认。当双方都收到对方的 SYN+ACK 数据段后，便都进入 ESTABLISHED 状态。图 10-43 显示了这种同时发起连接的连接过程，但最终建立的是一个 TCP 连接，而不是两个，这一点要特别注意。

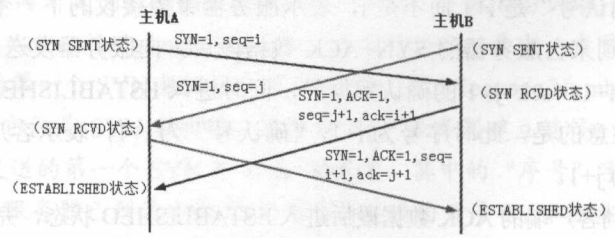


图 10-43　同时发起连接的 TCP 连接建立流程

从图中可以看出，一个双方同时打开的传输连接需要交换 4 个数据段，比正常的传输连接建立所进行的三次握手多交换一个数据段。此外需要注意的是，此时我们没有将任何一端称为客户或服务器，因为每一端既是客户又是服务器。

10.3.7 TCP 传输连接的释放

TCP 连接建立起来后，就可以在两个方向传送数据流。当 TCP 的网络应用进程再没有数据需要发送时，就可以发出关闭连接命令，释放连接。TCP 协议是通过发送 FIN 控制位置 1 的数据段来作为关闭传输连接的命令，关闭本端数据流的，但是本端仍然还可以继续接收来自对端的数据，直到对端也使用了同样的方法关闭那个方向的数据流，这时整个双方传输连接就彻底关闭了。

1. *单方主动关闭的 TCP 连接释放过程*

相对 TCP 传输连接建立的三次握手过程来说，TCP 传输连接的释放过程要稍微复杂一些，需要经过四次握手过程。这是因为服务器在收到客户端的 FIN 数据段后一般不会同时接收完数据，或者还有数据要向客户端发送，所以不会马上发送 FIN 数据段，只是先向客户端发送 ACK 对所接收的 FIN 数据段进行确认（此时客户端关闭了与服务器的传输连接），等数据全部接收完后再发送 FIN 数据段关闭本端与客户端的传输连接。

TCP 传输连接关闭的原则是：当一方完成它的数据发送任务后就可以发送一个 FIN 控制位置 1 的数据段来终止这个方向的数据发送（FIN 数据段的发送是由应用层调用 CLOSE 服务原语实现的）；当另一端收到这个 FIN 数据段后，必须通知它的应用层“对端已经终止了那个方向的数据传送”。TCP 连接释放的四次握手过程如图 10-44 所示。

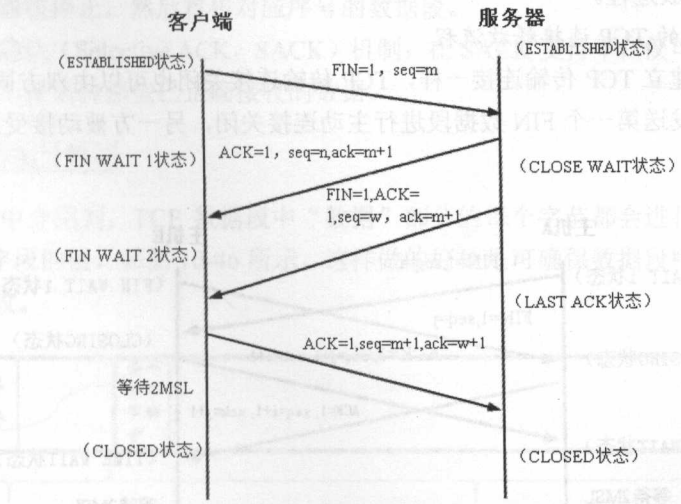


图 10-44 TCP 传输连接释放的四次握手过程

具体描述如下：

（1）一开始，通信双方都处于 ESTABLISHED（连接建立）状态。如果客户端认为数据全部

发送完了，想结束本次传输连接，则由应用层的对应应用进程调用 CLOSE 服务原语，然后向服务器发出一个 FIN 控制位置 1 的数据段（假设此数据段的序号为 m），客户端进入 FIN WAIT 1 状态，等待服务器的确认。

（2）服务器在收到客户端发来的 FIN 数据段后，确认客户端没有新的数据要发送了，向客户端发送一个 ACK 控制位置 1，“确认号”为 m+1 的数据段（假设此数据段序号为 w，服务器与客户端的数据段序号可以不一样），表示前面的数据已全部收到了，然后进入到 CLOSE WAIT（关闭等待）状态。与此同时服务器的 TCP 实体通知对应的应用层进程释放从客户机到服务器方向的传输连接，进入半关闭状态。但此时服务器仍可以向客户端发送数据段，客户端也可以接收来自服务器的数据。而且这可能要持续一段时间，直到服务器的数据也全部发送完。

（3）当客户端收到服务器的 ACK 数据段后便进入到了 FIN WAIT 2 状态，进一步等待服务器发出连接释放的数据段。

（4）当服务器接收完客户端的全部数据或者发送完全部向客户端传输的数据后，其对应的应用进程也会通知 TCP 实体释放此方向的 TCP 传输连接，向客户机发送 FIN 控制位置 1，ACK 控制位置 1，ack=m+1（假设此时的数据段序号已变为 w）的确认数据段，这时服务器进入 LAST ACK（最后确认）状态，等待客户端的确认。

（5）客户端在收到服务器的 FIN+ACK 数据段后，向服务器发送一个 ACK 控制位置 1，ack=w+1，序列号为 m+1 的数据段，进入到 TIME WAIT 状态。但此时 TCP 连接还没有释放，必须等待 2MSL 时间（RFC 793 建议设 MSL 为 2 分钟）后客户端才进入到 CLOSED 状态，彻底释放了 TCP 连接。

（6）服务器在收到客户端发来的 ACK 数据段后也进入 CLOSED 状态，彻底释放连接，完成整个 TCP 传输连接释放过程。

2. 双方主动关闭的 TCP 连接释放流程

与可以双方同时建立 TCP 传输连接一样，TCP 传输连接关闭也可以由双方同时主动进行（正常情况下都是由一方发送第一个 FIN 数据段进行主动连接关闭，另一方被动接受连接关闭），如图 10-45 所示。

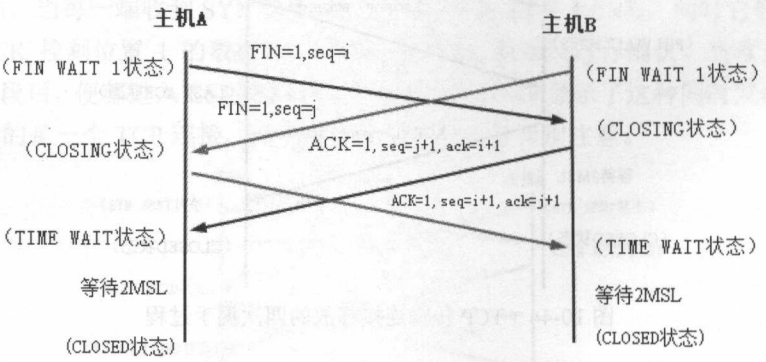


图 10-45　同时主动关闭 TCP 连接的流程

具体描述如下：当两端对应的网络应用层进程同时调用 CLOSE 原语发送 FIN 数据段执行关闭命令时，两端均从 ESTABLISHED 状态转变为 FIN WAIT 1 状态。任意一方收到对端发来的 FIN 数据段后，其状态均由 FIN WAIT 1 转变到 CLOSING 状态，并发送最后的 ACK 数据段。当收到最后的 ACK 数据段后，状态转变为 TIME_WAIT，在等待 2MSL 后进入到 CLOSED 状态，最终释放整个 TCP 传输连接。

10.4 TCP 的可靠传输机制

我们在本章前面一直在说 TCP 是一个可以提供可靠数据传输的传输层协议，那么它到底是怎样来保障可靠传输的呢？本节就来进行具体的分析。

在 TCP 可靠传输方面，主要采用了以下 4 种机制：

- 字节编号机制：TCP 数据段以字节为单位对数据段中的“数据”部分进行一一编号，确保每个字节的数据都可以有序传送和接收。
- 数据正确接收确认机制：这里面又包括两种措施：一是通过“确认号”字段获知对方已连续正确接收的数据段；二是通过“数据偏移”字段使在 TCP 数据分段重组时能准确地去掉除第一个数据分段外的 TCP 协议头部，再借助“序号”字段即可实现正确的数据段重组。
- 超时重传机制：在 TCP 中有一个重传定时器（Retransmission Timer，RTT），在发送一个数据段的同时也启动了该定时器。如果在定时器过期之前该数据段还没有被对方确认，则定时器被停止，然后重传对应序号的数据段。
- 选择性确认（Selective ACK，SACK）机制：在 SACK 支持下，仅可以重传缺少部分的数据，而不要重传那些已正确接收的数据。

10.4.1 TCP 字节编号机制

在 10.3.2 节中介绍到，TCP 数据段中“数据”部分的每个字节都会进行编号，第一个字节的编号是“序号”字段的值，如图 10-46 所示。这样做的好处是可确保数据段中的每个字节都可以有序地传送和被接收。

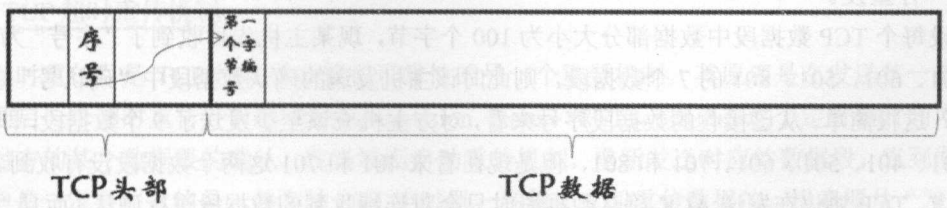


图 10-46 TCP 数据段中“序号”字段对应的字节编号

当接收端收到 TCP 数据段后，根据其中的“序号”字段以及去掉 TCP 报头后剩余部分的数据

字节数就可以得出要向对端发送 ACK 确认数据段时的“确认号”字段值，它就是先以“序号”字段值作为起始字节编号，然后依次对每个字节进行编号计算，直到最后一个字节，然后在最后一个字节的编号基础上加 1。如果一个 TCP 数据段被分段了，则下一个数据分段中的“序号”字段值是紧接着上一个数据分段“数据”部分最后一个字节的编号。在接收端进行数据段重组时，除第一个数据分段外，其他所有数据分段在接收端重组时都会去掉 TCP 报头。

如一个数据段“序号”字段值为 101，该数据段中“数据”部分所携带的是 100 个字节，则本数据段“数据”部分的第一个字节的序号是 101，最后一个字节的序号是 200，这样如果接收端收到这个数据段要进行确认的话，则在 ACK 确认数据段中的“确认号”字段值就是 201（200+1）。

另外，如果一个数据段在传输过程中被丢失了或者只正确接收了其中部分的数据，则接收端在向发送端发送 ACK 确认数据段时就不会包含整个该数据段，即整个该数据段以及后面所有已发数据段的内容都将由发送端重发，即使下一个或多个数据段已正确接收了。当然，这是在没启用后面将要介绍的“选择性确认”机制时。

10.4.2 TCP 正确接收确认机制

TCP 数据段发送完后需要等到对方发回一个包括该数据段确认信息的 ACK 数据段后才会确认对方已正确接收了该数据段，才会从自己的缓存中清除原来缓存的这部分数据，这就是 TCP 协议的“正确接收确认机制”。它包括以下两个方面的具体措施：①通过“确认号”字段确定对方已正确接收的数据段；②通过“数据偏移”字段实现无误数据段重组。

1. 通过“确认号”字段确定对方已正确接收的数据段

TCP 协议头部的“确认号”字段值表明了接收端已正确接收的连续数据段序号。对接收端已确认正确接收的数据从缓存表中清除，接收端没确认的数据在超时后从该数据段开始后面的所有已发数据段进行重发，不管某数据段是否已正确接收，具体特点如下：

- TCP 通常不会针对单个数据段进行确认，而是一次性对多个连续数据段进行确认，即凡是数据段“序号”字段值在所收到的 ACK 数据段中的“确认号”之前的数据段均已正确接收。
- 接收端仅对连续接收的数据段进行确认，不连续的数据段将先缓存。发送端在超时定时器过期后，对从第一个没收到 ACK 确认数据段开始后面所有已发的数据段全部进行重发。

假设每个 TCP 数据段中数据部分大小为 100 个字节，现某主机先后收到了“序号”为 1、201、101、301、601、501、801 的 7 个数据段，则此时该主机发送的确认数据段中“确认号”字段值是多少呢？这很简单，从已接收的数据段序号来看，对方主机应该至少发送了 9 个数据段，即 1、101、201、301、401、501、601、701 和 801，但是现在看来 401 和 701 这两个数据段没有收到。而前面又已知道，TCP 协议在发送 ACK 确认数据段时只会对连续收到的数据段进行确认，而且“确认号”为连续收到的数据段中“序号”最大的那个。在本示例中，连续收到了 1、101、201、301 这 4 个数据段，所以发送 ACK 确认数据段时的“确认号”值就是 301。

联想一下，如果主机收到了后面全部的 8 个，而 1 号数据段没有收到，情况会怎么样呢？很显然，这时主机不会像现在一样向对方发送 ACK 确认数据段，因为虽然后面全部 8 个数据段都收到了，但第一个数据段没有收到，仍是不连续的，所以既不会向对方发送 ACK 数据段，也不会把这些数据段交给设备功能模块来处理。当然这里仅指在没有启用“选择性确认”功能时。

【说明】TCP 头部的 ACK（确认）控制位指示 TCP 数据段中的“确认号”字段是否有效，占 1 位。仅当 ACK 位置 1 时（表示为 ACK 确认数据段）才表示“确认号”字段有效。

2. 通过“数据偏移”字段实现无误数据段重组

如果某个 TCP 数据段在发送时因超过 IP 协议对所接收的数据段大小的限制时就要进行分段了，这时每个 TCP 分段都会带上一个 TCP 协议头，到了接收端时只会保留第一个分段的 TCP 协议头，其他分段的 TCP 协议头都会去掉。但 TCP 协议头因为可以带上“可选项”字段，所以每个 TCP 数据分段的头部长度可能不一样。另外，因为 TCP 数据段是以数据流方式进行传输的，而不是像 UDP 协议那样以整个报文方式进行传输，这样就不能从数据流中区分哪部分是 TCP 协议头，哪部分是“数据”字段部分。这样就遇到问题了，那就是在接收端如何确定每个 TCP 数据分段要去掉的 TCP 协议头长度，实现各数据分段中真正“数据”部分的重组呢？

TCP 协议设计之初当然考虑到了这个问题，自然有相应的解决措施，那就是在每个数据段或分段头部长度都有一个“数据偏移”字段，如图 10-47 所示。这个字段是指数据段中“数据”部分与分段开始处以 4 个字节为单位的偏移量（占 4 位，即最长为 15*4=60 个字节），也就是指定了每个 TCP 数据段或数据分段头部的大小，这样就可以非常准确地去掉那些除第一个分段外的所有其他分段的 TCP 协议头部了，然后再根据每个数据段中“序号”字段由小到大的顺序把这些纯数据部分进行组装即可。

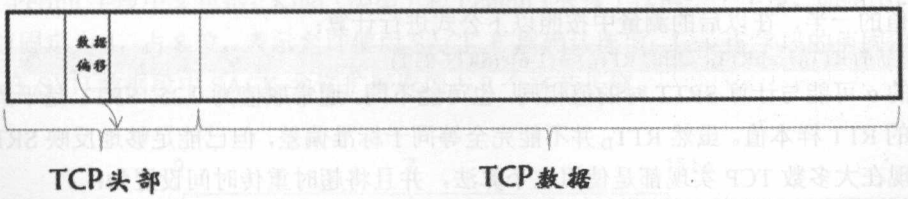


图 10-47 TCP 协议头部的“数据偏移”字段

10.4.3 TCP 超时重传机制

“超时重传”是 TCP 协议保证数据可靠性的另一个重要机制，其原理是在发送某一个数据段以后就开启一个超时重传计时器（Retransmission Timer，RTT），如果在这个定时器时间内没有收到来自对方的某个数据段的确认，发送端则启动重传机制，重新发送对应的数据段，直到发送成功为止。但是一定要注意的是，并不是 RTT 定时器一到就会立即重传数据的，毕竟要从“发送窗口”缓存中找到对应的数据段，然后安排重新发送都是需要时间的，所以实际上超时重发的时间间隔（Retransmission Time Out，RTO）要大于 RTT 值。

1. SRTT 的计算

这里涉及到两个重要问题：一是如何确定 RTT 值，另一个是如何计算 RTO 值。表面上看起来 RTT 值容易确定，因为它就是一个数据段往返发送端和接收端的时间总和，但在 TCP 通信中，中间可能经过了多个性能不一样的网络，而且不同时刻网络的拥塞程度都可能不一样，这就造成不同数据段的 RTT 时间并不一致，甚至波动非常大。但 TCP 必须适应这种情况，必须能动态地跟踪这些变化并相应地改变其超时重传时间。

正因为 RTT 值不是固定的，所以就有了一个平滑 RTT（SRTT）的概念，就是在充分考虑历史 RTT 值的情况下所设计的一个 RTT 值计算公式。在最初的 RFC 793 中，SRTT 的计算公式如下：

SRTT（新的 SRTT）=αSRTT（旧的 SRTT 值）+(1-α)RTT（新的 RTT 样本值）　　①

SRTT 的初始值就是第一个 RTT 值，这里的 α 是一个平滑因子，它决定了旧的 SRTT 值所占的权重，$0 \leqslant \alpha < 1$，RFC 2988 推荐的 α 典型值为 0.125。如果 α 很接近 1，则表示新的 SRTT 值与旧的 SRTT 值很接近，变化不大，相反则变化比较大。显然用这种方法计算得出的各个时刻的 SRTT 值更加平滑，更加接近当时的网络环境。

2. RTO 的计算

虽然通过公式①可以计算出 SRTT 值了，但是 RTO 的得出仍不是一件简单的事，因为到底是 RTT 定时器到后就重传，还是要再等待多长时间才重传，这是必须要考虑的。正常情况下，TCP 使用 βSRTT 作为重传超时间隔（$\beta > 1$），而且最初的值总为 2（也就是两倍于 SRTT 时间后还没收到对应数据段的确认才重传该数据段）。

但经验表明采用常数的 β 值不够灵活，于是 1988 年 Jacobson 提出使用平均偏差作为标准偏差（就是 βSRTT）的估计算法，要求维护另一个被平滑的偏差 RTT_D。当一个确认数据段到达时可以得出 SRRT 和新的 RTT 样本值之间的偏差 RTT_D=(SRTT-RTT)。第一次测量时，RTT_D 为测量到的 RTT 样本值的一半，在以后的测量中按照以下公式进行计算：

RTT_D（新的 RTT_D）=αRTT_D（旧的 RTT_D）+(1-α)×(SRTT-RTT)　　②

这里的 α 可能与计算 SRTT 时的值相同，也可能不同，通常取值为 0.25，SRTT 是平滑 RTT 值，RTT 是新的 RTT 样本值。虽然 RTT_D 并不能完全等同于标准偏差，但已能足够地反映 SRTT 值的动态变化。现在大多数 TCP 实现都是使用这个算法，并且将超时重传时间设置为：

RTO=SRTT+4×RTT_D　　③

这里有一个重要的问题，那就是对于重传的数据段如何计算其 RTT 值。假设发送端发送了一个数据段，可是在重传定时器内没有收到该数据段的确认数据段，于是重新发送该数据段，可是过了一段时间，发送端又收到了该数据段的确认数据段。这时发送端就迷糊了，这个确认数据段到底是对原来发送的那个数据段的确认，还是对重发的数据段的确认呢？因为这两个数据段的序号是一样的，如果在上次发送数据段后没有再重新发送新的数据段的话，接收端返回的确认数据段中的“确认号”都可能一样。如果收到的确认数据段是对重传数据段的确认，但却被发送端认为是对原数据段的确认，则这样计算出的 SRTT 和 RTO 值肯定会偏大，否则会偏小。为此，一位发现这个问题的无线电爱好者 Karn 提出了一个建议，就是在计算加权平均 RTT 时，只要数据段被重发了，就不采用其往返时间作为计算 SRTT 和 RTO 的样本，这样得出的加权平均 SRTT

值和 RTO 值就比较准确了。

10.4.4 TCP 选择确认机制

在上节介绍的 TCP 重传机制中，如果在重传定时器超时后仍没收到一个数据段的确认，则可能会重传对应序号后面的所有数据段，因为后面的这些数据段均暂时不会被确认，这明显大大降低了 TCP 数据传输的性能。

同样假设每个数据段大小为 100 个字节，假设接收端已收到 1、101、201、401、501 这 5 个序号的数据段，其中 301 号数据段没有收到。按照 10.4.2 节介绍的确认机制，虽然 401 号和 501 号这两个序号的数据段均已收到，但就因为 301 号这个数据段一直没有收到，仍然不能向发送端确认。这时在某个时间上，如果 301 号数据段的重传定时器超时了，则发送端肯定会重传这个数据段，但毕竟重传的 301 数据段到达对端也是需要时间的，在这个 301 号数据段的重传过程中，401 号甚至 501 号数据段的重传定时器可能超时，这时发送端可能又对 401 号和 501 号这两个数据段进行重传。这样的重传显然是不必要的，也会造成接收端重复接收 401 号和 501 号这两个数据段。

为了避免这种现象的出现，在 RFC 3517 中就出现了一种称为“选择性确认”（SACK）的机制，在 TCP 数据段格式头部的“可选项”字段中添加了一个代表支持 SACK 的选项。但这个选项在不同的数据段中有不同的字段名称和不同的含义。首先必须在建立 TCP 传输连接时的 SYN 数据段中包含 SACK-Permit（SACK 允许）字段选项，表示在今后的传输中希望收到 SACK 选项。然后在其他的数据段中需要包含 SACK 字段，包含本端要告诉对方已接收到的不连接数据段。原来的“确认号”字段同样有效，但此时的 SACK 扩展选项也仅在确认数据段（ACK 字段值为 1 时）中才有效。

SACK-Permit 字段中又包括：Kind（类型）和 Length（长度）这两个子字段，如图 10-48 所示。Kind 字段值固定为 4，占 8 位，表示允许使用 SACK 扩展确认选项；Length 字段的值固定为 2，占 8 位，表示在 SYN 数据段中 SACK 允许扩展选项占 2 个字节长度。

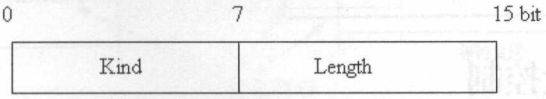


图 10-48 SYN 数据段中的 Sack-Permitted 选项格式

在其他非 SYN 数据段的 SACK 字段中包括 Kind、Length，以及各不连续数据段的起始字节号和结束字节号，如图 10-49 所示。Kind 字段固定值为 5，占 8 位，表示这是非 SYN 数据段的 SACK 选项；Length 字段值可变，占 8 位，以字节为单位表示 SACK 扩展选项的长度。后面是 n 个标识不连续数据段起始序号和结束序号的部分，每个序号占 32 位。

因为 TCP 数据段头部的“可选项”字段最长为 40 个字节，Kind 和 Length 这两部分一共占了 2 个字节，而表示一个不连续数据段的起始序号和结束序号要占用 8 个字节（因为一个序号是 4 个字节长度），所以实际上在一个数据段中最多可以标识 4 个不连续数据段，因为标识 4 个数据段的

起始序号和结束序号一共要用到32个字节，加上前面的Kind和Length这两部分就有34个字节了。而要标识5个数据段的起始序号和结束序号则还要8个字节，超出了“可选项”字段40个字节的限制范围。所以在Length子字段中最大值其实就是34。

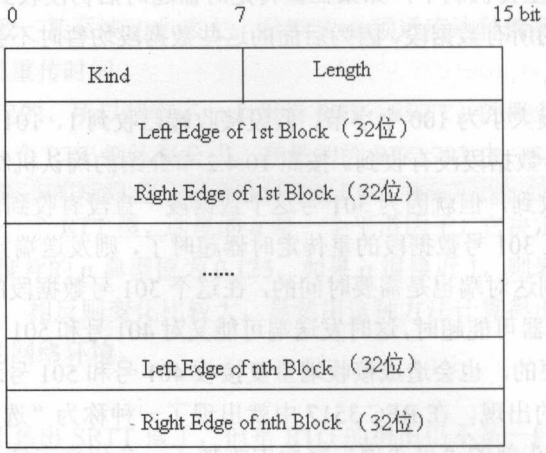


图10-49　其他数据段中的Sack选项格式

发送端通过识别接收端返回在ACK确认数据段中的SACK扩展选项就可以得知接收端已收到了哪些不连续序号的数据段，这样发送端就可以不再发送这些数据段，而只发送已丢失（发送端已发送，且在重传定时器超时后仍看到接收端没有收到）的数据段。

还是以前面的例子进行介绍，假设接收端已收到1、101、201、401、501这5个序号的数据段，在发送“确认号”为301的确认数据段时，在SACK扩展选项中标记401（起始序号为401，结束序号为500）和501（起始序号为501，结束序号为600）这两个不连续的数据段。这时发送端就知道不需要再发送401号和501号这两个数据段了，只需发送301号数据段。大大节省了网络资源，也提高了数据传输效率。

10.5　TCP的流量控制

“流量控制”和下节将要介绍的“拥塞控制”是两个不同的概念，当然它们之间还是有些关联的地方。“流量控制”是基于通信双方的数据发送和接收速率匹配方面考虑的，其最终目的就是不要让数据发送得太快，要让接收端能够来得及接收，是一个链路两端的点对点行为。而“拥塞控制”则是基于网络中各段链路的带宽和中间设备的数据处理能力方面而考虑的，不要使网络中出现数据传输阻塞，也就是不要出现发送端发送的数据大于接收端数据处理能力的情形，是一个端到端的行为。但流量控制又可能对拥塞控制有所帮助，其实解决好TCP连接所经过的所有链路的流量控制问题，也就基本上解决了拥塞控制问题了。本节先来介绍TCP的流量控制方案。

10.5.1 TCP 流量控制

在数据传输中可能会出现这种情形，那就是发送端一次发送的数据太大，接收端来不及处理，同时可用的缓存又放不下，这时这部分数据只能在接收端被丢弃了。这样就大大影响了数据传输的可靠性和传输效率，因为这么大的数据被丢弃后又得重传，又得消耗大量的设备和带宽资源。

为了解决这个问题，在 TCP 协议的头部加了一个名为“窗口大小”的字段，它是用来告诉对方本端下次可接收数据的最大大小（以字节为单位），这样对方在向本端发送数据时就会自动把要发送的数据大小调整在“窗口大小”字段值的范围内了，可以确保发送的数据不会在接收端被丢弃。

但是也要注意，实际发送的数据大小虽然一定不能大于所接收到的对方发来数据段中的“窗口大小”字段值大小，但并不一定就等于这个“窗口大小”字段值，因为这还要受本端的“发送窗口”大小影响，而且可能是在不断变化的。这就是我们所说的“滑动窗口”的含义。“发送窗口”大小又与缓存中所缓存的数据大小有关，所缓存的数据包括两个部分：一是来自对方但本地设备还来不及处理的数据，二是自己向对方发送但还没有收到对方对这些数据段的确认的数据。也就是说，缓存中可能会同时包含来自对方的数据以及自己原来发送的数据。

1. 正常情况下的滑动窗口流量控制机制

如图 10-50 所示是一个滑动发送窗口示例，图中的序号为每个数据段的序号，也就是对应数据段的第一个字节序号。现假设每个字段的大小为 100 个字节（这仅是为了方便介绍，实际上的数据段大小一般都在千个字节以上），假设物理“发送窗口”大小为 500 个字节（这也是为了方便介绍，实际上的窗口大小远大于这个值）。

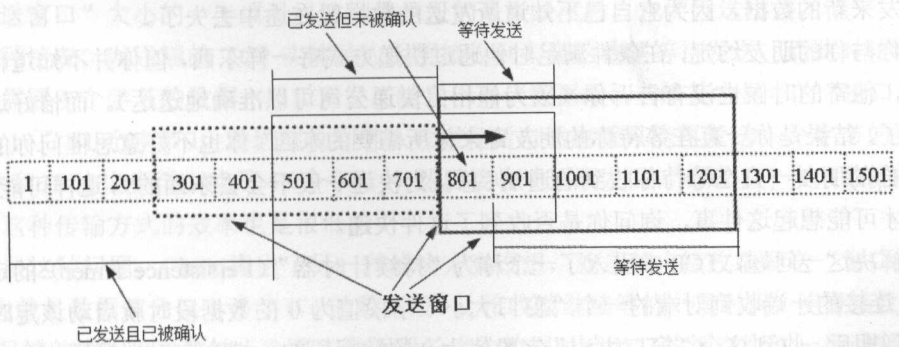


图 10-50 TCP 滑动窗口示例

（1）假设现在发送端收到了接收端发来的一个确认数据段，“确认号”为 301，“窗口大小”为 500，表示可以连续发送 5 个数据段（起始序号为 301）。左边那个虚线“发送窗口”代表的就是后面发送的 301、401、501、601、701 这 5 个数据段，此时因为已达到了对方窗口大小值，不能继续发送了，需要停下来等待对端的确认。

（2）如果在发送数据前又收到了来自对方的一个“确认号”为 501 的确认数据段（ACK 控制位为 1 的数据段，下同），则表示接收端已正确接收了 301 和 401 这两个数据段，即从“发送窗口”

中删除这两个数据段，窗口向前滑动 200 个字节（也就是 301 和 401 这两个数据段的大小），移到了图中细实线“发送窗口”位置。其中 501、601 和 701 这 3 个数据段是原来已发送但没收到确认的缓存数据段，801 和 901 这两个数据段才是真正要等待发送的。

假设以上返回的确认数据段中的“窗口大小”为 400（这里就起到了流量控制的作用），理论上说发送端可以一次性连续发送 4 个数据段。但是因为“发送窗口”中缓存了原来已发送的 501、601 和 701 这 3 个数据段（300 个字节），于是此次只能发送 100 个字节了，即只能先发送后面的 801 号数据段了。

（3）如果此后又收到了一个“确认号”为 801 的确认数据段，同时“窗口大小”字段值又为 500 了（这里也起到了流量控制的作用），则发送端知道接收端已收到了 701 号及以前所有的数据段了，于是从“发送窗口”中删除这些缓存的数据段，此时实际上缓存的数据段仅为 801 号数据段。但原来的 901 号数据段已在“发送窗口”中，等待发送，所以此时“发送窗口”只需再继续向前移 300 个字节（总共发送 400 个字节），如图中的粗实线“发送窗口”，继续发送 901、1001、1101、1201 这 4 个数据段。

2. *存在数据丢失情况下的流量控制*

如果在数据传输过程中有一个或多个数据段丢失，则发送端接收不到对这些数据段的 ACK 确认数据段，这时可以通过上节介绍的超时重传来解决。现在要探讨的一个问题是，如果某个时间对端发送的数据段显示“窗口大小”字段值为 0，这时发送端自然不能再发送数据了，只好等待对方发来一个“窗口大小”字段值不为 0 的数据段。可是对端发来的这个数据段在传输过程中丢失了，这时对于发送端来说，一直在等待来自接收端“窗口大小”字段值非 0 的数据段，而接收端又在等待发送端发来新的数据，因为它自己不知道所发送的数据段在途中丢失了。

就像你与你的朋友约定，在条件满足时他通过快递方式寄一样东西，但你并不知道他具体是什么时候寄，他寄的时候也没有告诉你（因为他相信快递公司可以准确地送达）。而恰好这件快递在途中丢失了，结果是你一直在等待你的朋友寄来你所需要的东西（你也不好意思催问你的朋友是否寄了），而你的朋友一直在等待你收到的通知（总认为快递一般不会丢东西的）。这样可能等了好久，你的朋友才可能想起这件事，询问你是否收到了这件快递。

为了解决这一问题，TCP 中引入了一个称为“持续计时器”（Persistence Timer）的定时器。它是在 TCP 连接的一端收到对端的一个“窗口大小”字段值为 0 的数据段时就启动该定时器；在这个定时器到期后，收到这个“窗口大小”字段值为 0 的数据段的一端会向对端发送一个非常小的探测数据段（一般仅携带 1 个字节的数据），这时对端在收到这个探测数据段后会返回一个确认数据段。如果在确认字段中的“窗口大小”字段值仍为 0，则发送端重启上面的“持续定时器”，否则结合确认数据段中的“窗口大小”字段值和当前可用的“发送窗口”大小发送相应字节的数据，解除以上这种双方持续等待的局面。

10.5.2 传输效率的考虑

我们在前面已经说了，TCP 数据段中只对“数据”部分的字节进行编号，原因是在对方传输

层解封装后得到，并且所需要的只是“数据”部分内容，这样就可以很容易地知道所接收的数据是否连续；否则如果加上 TCP 协议报头，以及传输到网络层的 IP 协议头和传输到数据链路层的报头，数据部分的编号就不可能连续，况且不同数据段的报头大小都可能不一样，因为还有“可选项”部分。

对于大型的数据传输，这些报头加起来也才几十个字节，对于一个数据段动辄上千个字节来说，这些协议头部分显然是可以忽略不计的，也就是说传输效率是非常高的。但是我们要考虑到各方面的应用，如一些交互式应用中，每次传输的数据部分可能仅 1 个或几个字节，如果为每个这样的数据传输一次，显然传输效率是很低的，因为用几十个协议头最终传输的有用数据仅几个字节。这类情况还会在发送确认数据段时经常发生，因为如果单单是用来确认的数据段，里面的数据量是非常小的，也正因为如此，所以通常是在捎带大量数据的数据段中把 ACK 控制位置 1，同时起到确认的作用。

这时就涉及到什么时候把这些数据组装成一个数据段进行传输的问题。这里涉及到两个非常重要的算法：Nagle 算法和 Clark 算法。

Nagle 算法规定：如果数据每次需要发送 1 个字节，则发送端首先只发送第一个字节，其余字节先缓存起来，直到发送出去的那个字节被确认为止。然后再次把所有缓存起来的数据放在一个 TCP 数据段中发送出去，并同时缓存这些字节，直到发送出去的字节全部被确认后再从缓存中清除。在这种算法下可以大大提高传输效率，因为不会因 1 个字节而用了几十个协议头来携带。试想一下，如果一个用户的键盘输入很快，而网络很慢，这时就会把用户的许多次输入字符组装成一个数据段，然后一次发送出去，大大减少了所占用的网络带宽。Nagle 算法还规定，当要发送的数据已达到“发送窗口”大小的一半或者已达到数据段的最大长度时，也要发送出去。

还有一种情况，发送端的 TCP 实体从应用层接收到大量的数据等待发送，而此时接收端的应用却每次只需要 1 个字节的数据，然后向发送端发出确认数据段，当然其中显示的“窗口大小”字段值就只是 1 了（1 个字节）。发送端收到确认数据段后自然也就只能再发送 1 个字节的数据（但其中的 TCP 和 IP 协议头至少就是 40 个字节），然后接收端再次发回针对这 1 个字节数据的确认数据段。显然这种传输方式的效率也是很低的。

为了解决这种问题，Clark 提出了另一种解决方案设想，就是禁止接收端发送“窗口大小”字段值为 1（也就是数据部分仅 1 个字节）的数据段。让接收端继续等待一段时间，使得接收端的“接收窗口”有足够的空间可以容纳一个最大数据段长度的数据时或者它的缓存空间一半已空时（取两者的最小值）才发送确认数据段。这时“窗口大小”字段值就肯定不是 1 了，这样发送端就可以一次性发送更多的数据，提高了传输效率。

10.6 TCP 的拥塞控制

TCP 协议中最复杂的功能可能就是拥塞控制功能了，因为两端经过的网络数量、网络类型、网络性能都可能千变万化，而这些因素又紧密地影响着 TCP 连接拥塞控制的复杂性和难度。TCP

的拥塞控制主要依靠 TCP 连接双方商定的协议减少数据的发送来实现。

10.6.1 TCP 拥塞控制概述

拥塞控制（Congestion Control）顾名思义就是要控制“网络拥塞”的出现。那什么情况下会出现网络拥塞呢？简单地说就是网络中各个部分输入的流量大于输出的流量时就会出现网络拥塞。打一个比较类似的比喻（之所以说是“类似”，因为它更像“流量控制”类的比喻），就像在公路上发生塞车的一个根本原因就是出口道路太窄，而入口道路又比较宽，导致同一时间驶入的车辆数大于驶出的车辆数，最终使得这条路上排队挤满了各种车辆，车辆的行驶速度自然会降下来。最坏的情况是什么，那就是完全堵死，车辆根本出不去也进不来。这就类似网络中通常所说的“死锁”现象。

当然，道路上发生塞车的原因不仅只有上面所说的那一个，还有像上游驶入的车辆太多，中间或下游正在进行车辆检查，中间某些车辆行驶速度太慢或者发生了交通事故而占用了部分道路等。就像道路发生塞车的原因多种多样一样，网络上发生拥塞的原因也可能是多方面的，而且网络结构越复杂，发生拥塞的原因也可能越复杂。如 TCP 连接的整个链路中有结点设备的缓存空间太小，数据转发能力太低或者某段链路带宽太小，当然也可能是对端数据能力低等都可能引起网络拥塞。而且往往是多个因素同时存在的，所以处理拥塞控制问题不能简单地针对某一方面来加以解决，必须从全局角度来寻找解决方案，否则可能不仅不能解决拥塞现象，反而会形成新的瓶颈，使网络更加拥塞。

如你单方面地提高中间路由器结点的数据转发能力，而忽视了所经路径上各段链路的带宽，结果你提高了路由器转发性能，数据转发速度是提高了，但也同时造成了在链路上排队前行的数据增多了，不仅没有解决网络拥塞问题，反而使网络拥塞得更严重。再如你单方面地增加路由器的缓存能力，但却没有同步考虑路由器的数据转发能力和链路的带宽，结果虽然可以使更多的数据在路由器上暂时缓存，但在缓存中排队的数据所需等待的时间会更长，因为队列比以前更长了，结果会因为超时而重传这些数据。重传数据越多越增加了网络负荷，最终导致拥塞更严重。就像我们在车站买票时，如果前面售票员的售票速度没有提高，仅增加了用于排除等候的空间，让队列可以排得更长，是不能解决长时间排队买票难题的，反而使排在后面的人买到票的时间会更长，使得售票厅排队买票的人员更加拥挤。

如果我们把整个网络有效处理负荷的能力称为“吞吐量”（Throughout），而把网络中发送端输入的负荷称为“输入负荷”（Input Load），则理想情况下它们之间有如图 10-51 所示的线性关系（吞吐量是呈 45°斜线上升的）。

从图中可以看出，在开始时，网络系统的吞吐量随着输入负荷的增加呈同步提高态势（两者的提高量是一样的），但当到了一定时期（在输入负荷达到了网络的最大吞吐量时），输入负载再怎么增加，网络系统的吞吐量都不再提高了，而是保持在一个不变的水平。这时就出现了轻度拥塞现象。就像一条河流最初是没有水的，所以当上游河流放水时它的水速会随着上游流入水的水速提高而提高，但当这条河流满了，也就是到了它的最大负荷水平，它的水速不会再有提高了，尽管上游流入

的水速仍在提高。那么从上游新增加的那部分水到哪里去了呢？肯定是被溢出了。相当于网络到达最大吞吐量时从发送端发来的超出吞吐量那部分的数据会被丢弃一样。

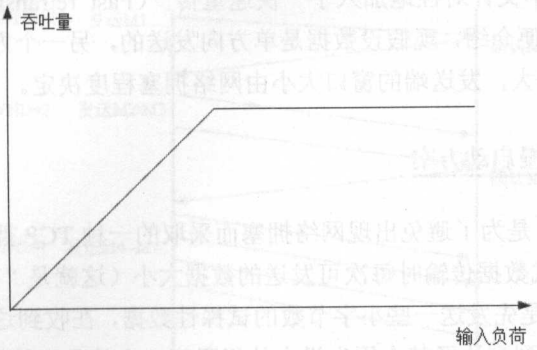


图 10-51　理想情况下的吞吐量与输入负荷之间的关系

但图 10-51 只是一种理想情况，而实际上网络系统吞吐量与输入负荷之间的关系永远不会是线性关系。因为网络中不可能完全是理想状态，所以一开始“吞吐量”和“输入负荷”之间的关系就不是呈线性比例的。当输入负荷到达或接近最大吞吐量时，如果继续提高输入负荷，此时网络系统的实际吞吐量是不会保持在最大吞吐量水平的，而是呈现下降趋势的。如果此时输入负荷继续增加，最终可能会导致网络系统的吞吐量下降到 0，出现完全死锁状态，如图 10-52 所示。

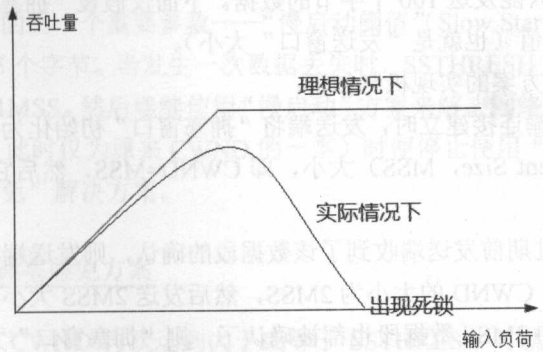


图 10-52　实际情况下的吞吐量与输入负荷之间的关系

为了避免这种现象的发生，就需要对网络中的流量进行控制，这就是后面各小节将要介绍的传输层 TCP 协议所采用的拥塞控制方法。因为我们知道，TCP 是一种可靠的传输层协议，它必须充分考虑到网络中出现拥塞的可能性并提供各种应对之策。

尽管我们知道了发生拥塞的原因，也知道了网络系统吞吐量与输入负荷之间的关系，但要真正设计一套有效的拥塞控制方案却不是一件容易的事，因为拥塞的出现和所引起的因素都是动态的，是在不断发生变化的。但发生网络拥塞有一个最根本的征兆就是出现了数据的丢失。于是，为了防

止网络的拥塞现象，出现了一系列的 TCP 拥塞控制机制。最初是由 V.Jacobson 在 1988 年的论文中提出的 TCP 的拥塞控制，由“慢启动”（Slow start）和“拥塞避免”（Congestion avoidance）组成，后来在 TCP Reno 版本中又针对性地加入了“快速重传”（Fast retransmit）和“快速恢复”（Fast Recovery）算法。为了方便介绍，现假设数据是单方向发送的，另一个方向只传送确认数据段，而且假设接收端的窗口足够大，发送端的窗口大小由网络拥塞程度决定。

10.6.2 TCP 拥塞控制的慢启动方案

慢启动（Slow start）是为了避免出现网络拥塞而采取的一种 TCP 拥塞初期预防方案。其基本思想就是在 TCP 连接正式数据传输时每次可发送的数据大小（这就是“拥塞窗口”（CWND）的含义）是逐渐增大的，也就是先发送一些小字节数的试探性数据，在收到这些数据段的确认后，再慢慢增大发送的数据量，直到到达了某个原先设定的极限值，也就是下面将要提到的“慢启动阈值”（SSTHRESH）。

在“慢启动”拥塞解决方案中，发送端除了要维护正常情况下根据接收端发来的“窗口大小”字段值而调整的“发送窗口”外，还要维护一个“拥塞窗口”（Congestion Window，CWND），它是为了避免发生拥塞而设置的窗口。最终允许发送的字节数是这两个窗口中的最小值。如从接收端返回的“窗口大小”字段值是 100，而当时设置的“拥塞窗口”大小为 50，则此时只能发送 50 个字节的数据。相反也一样，如从接收端返回的“窗口大小”字段值是 100，而当时设置的“拥塞窗口”大小为 200，则此时只能发送 100 个字节的数据。下面仅假设“拥塞窗口”总是小于从接收端返回的“窗口大小”字段值（也就是“发送窗口”大小）。

“慢启动”拥塞解决方案的实现机制如下：

（1）在一个 TCP 传输连接建立时，发送端将“拥塞窗口”初始化为该连接上当前使用的最大数据段（Maximum Segment Size，MSS）大小，即 CWND=MSS，然后它发送一个大小为 MSS 的数据段。

（2）如果在定时器过期前发送端收到了该数据段的确认，则发送端将“拥塞窗口”大小再增加一个 MSS，也就是此时 CWND 的大小为 2MSS，然后发送 2MSS 大小的数据。

（3）如果这次发送的 2MSS 数据段也都被确认了，则“拥塞窗口”大小再增加两个 MSS（此时相当于达到了 4MSS），依此类推。也就是每一次发送数据时的 CWND 和所发送的数据段大小都是前一次发送的两倍。

如图 10-53 所示是一个采用“慢启动”方案的示例，M 代表 MSS，就是“最大数据段大小”的意思。第一次发送一个 MSS（即 M1），收到 M1 的确认后再连贯发送两个 MSS（即 M2 和 M3），当收到 M3 的确认（因为后面的确认了的话，预示前面的都已正确接收，下同）后再发送 4 个 MSS（即 M4～M7），当收到 M7 的确认时，后面将发送 8 个 MSS（即 M8～M15），……。

“慢启动”方案的基本规律是：当“拥塞窗口”大小达到了 n 个 MSS 时，如果所有 n 个数据段都被及时确认，则新的“拥塞窗口”大小再增加 n 个 MSS，也就是新的“拥塞窗口”大小是旧

的“拥塞窗口”大小的 2 倍关系，即“拥塞窗口”大小与最大数据段大小（MSS）的关系是 2 的指数关系，即 1MSS，2MSSS，4MSS，8MSS，16MSS，32MSS，……。

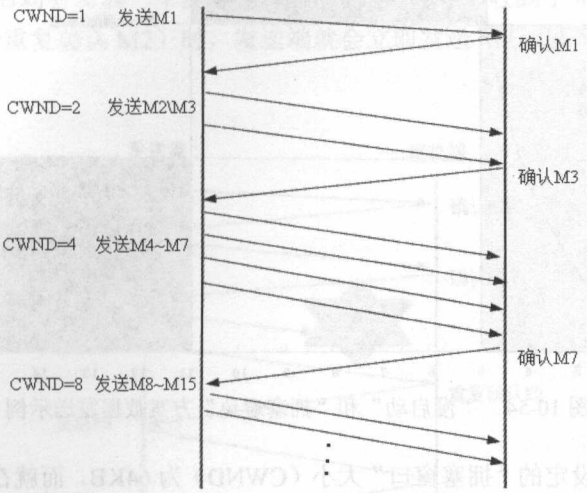


图 10-53 “慢启动”解决方案 TCP 数据发送示例

但是“拥塞窗口”不可能无限制地继续增大，即使 CWND 值仍是小于从接收端返回的“窗口大小”字段值，也有可能在某个时刻出现数据丢失的现象。这个“拥塞窗口”大小就是一个临界点，于是引入了“慢启动”方案的另一个重要参数——“慢启动阈值”（Slow Start Threshold，SSTHRESH），初始值为 64KB，即 65535 个字节。当发生一次数据丢失时，SSTHRESH 设为当前 CWND 的一半，而 CWND 又重新被置为 1MSS。然后继续使用“慢启动”方案来解决网络拥塞问题，不过当 CWND 再次增长到 SSTHRESH（此时仅为原来 CWND 的一半）时便停止使用“慢启动”方案，需要采用下节将要介绍的“拥塞避免”解决方案。

10.6.3 TCP 拥塞控制的拥塞避免方案

当本次发送数据时的 CWND 再次达到大于或等于 SSTHRESH 时就启动“拥塞避免”解决方案。它的基本思想是：在 CWND 值第二次到达 SSTHRESH 时，让“拥塞窗口”大小每经过一个 RTT（一个数据段往返接收端和发送端所需的时间）时间仅值加 1（即新的 CWND 只增加一个 MSS 大小，而不是原来 CWND 的翻倍），使其以线性方式慢慢地增大，而不是继续像“慢启动”方案中那样以指数方式快速增大。显然这种 CWND 增长速度明显要慢于“慢启动”方案中的 CWND 增长速度。当再次发生数据丢失时，又会把 SSTHRESH 减为当前 CWND 的一半，同时把 CWND 置为 1，重新进入“慢启动”数据发送过程，依此类推。

如图 10-54 所示是一个“慢启动”和“拥塞避免”这两种拥塞控制方案先后采用的控制示例。

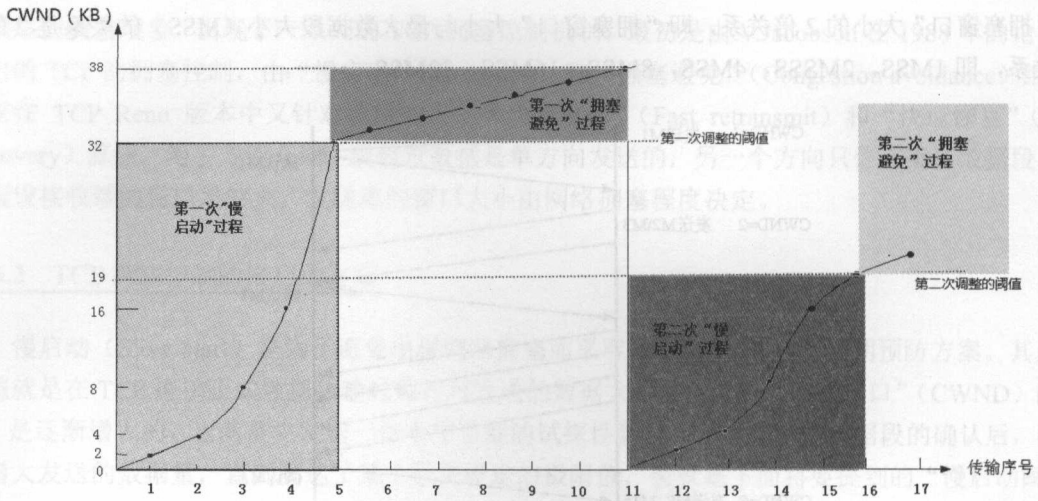


图 10-54　“慢启动”和“拥塞避免”方案数据发送示例

假设在某个时间所设定的“拥塞窗口”大小（CWND）为 64KB，而就在此后发生了数据丢失，没有收到后面发送的数据的确认，于是“慢启动阈值”（SSTHRESH）减为 CWND 的一半，即 32KB（即图中的“第一次调整的阈值”），同时把新的 CWND 设为 1MSS，从坐标原点开始重新采用“慢启动”方案进行数据发送；当达到了 SSTHRESH 值（即 32KB）时，采用“拥塞避免”方案进行数据发送，每个 RTT 时间 CWND 只增加一个 MSS。

假设在第 11 次传输过程中（此时 CWND=38KB）又发生了一次数据丢失，于是 SSTHRESH 再次降为当前 CWND（即 38KB）的一半（即 19KB），并且重新开始采用“慢启动”方案发送数据，直到到达新的 SSTHRESH 值（19KB），然后再从这点开始又采用“拥塞避免”方案发送数据，依此类推。

10.6.4　TCP 拥塞控制的快速重传/快速恢复方案

前面介绍的“慢启动”和“拥塞避免”是 1988 年提出的拥塞控制方案，1990 年又新增了两种拥塞控制方案，那就是此处要介绍的“快速重传”和“快速恢复”拥塞控制方案。

“快速重传”方案的基本思想是：当接收端收到一个不是按序到达的数据段时，TCP 实体迅速发送一个重复 ACK 数据段，而不用等到有数据需要发送时顺带发出确认；在重复收到 3 个重复 ACK 数据段后，即认为对应“确认号”字段的数据段已经丢失，TCP 不等重传定时器超时就重传看来已经丢失的数据段。为了进一步理解“快速重传”原理，现举如图 10-55 所示的一个示例。

假设每次只发送一个大小等于 MSS 的数据段，并假设第一次、第二次发送的 M1、M2 发送端都从接收端收到了确认，但第三次发送的 M3 在途中丢失了，而后面第四次发送的 M4 又收到了。正常情况下，接收端是不会再发送任何确认数据段的，直到收到 M3。而发送端此时并不知道 M3 丢失了，继续发送 M4，在接收端收到 M4 时，为了尽快通知发送端 M3 还没有收到，于是再次发

送一个 M2 确认数据段（其中的“确认号”字段值是 M3 的序号），相当于再次提醒发送端 M3 数据段没有收到。但此时发送端还不会重发 M3，希望再等等看，仍然继续依次发送 M5、M6，在接收端每收到一个数据段后均会发送一个重复 M2 确认。到了发送端收到了 4 个针对 M2 数据段的确认（一个确认 M2，3 个重复确认 M2）时，发送端就会立即发送 M3，尽管此时可能 M3 的重传定时器还没有过期。

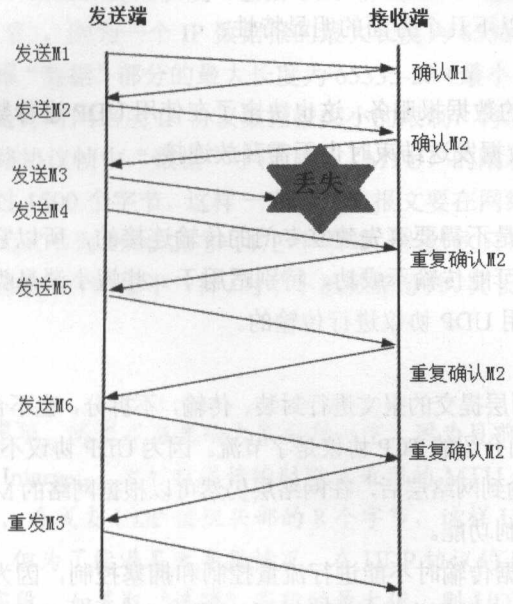


图 10-55 “快速重传”方案数据重传示例

在“快速重传”算法发送了看来已经丢失的数据段之后，“快速恢复”算法同时开始作用。“快速恢复”算法的基本思想是：在收到第 3 个重复 ACK 时，把当前 CWND 值设为当前 SSTHRESH 值的一半，减轻网络负荷负担，然后执行前面介绍的“拥塞避免”算法，使 CWND 值慢慢增大，以避免再次出现网络拥塞。

10.7 UDP 协议

在网络体系结构“传输层”中主要的传输层协议就是前面介绍的 TCP，毕竟它是可靠的传输层协议，可以提供可靠的数据传输。但是有一些网络应用，应用的连续性和可靠性可能更重要，如视频会议、语音电话，其中丢失一部分数据的影响并不大，用户通过上、下部分的内容可能“猜”得出来，甚至这部分影响很难看得出来，但如果出现传输中断或者延迟严重可能就无法接受了。于是就出现了一种不面向连接的传输层协议——UDP 协议（User Datagram Protocol，用户数据报协议）。

10.7.1 UDP协议概述

UDP是一种无连接传输层协议，无需像TCP协议那样需要服务器端监听，也不必等待客户端与服务器端建立连接后才能通信，当然最后能否把数据传输成功，UDP是不能保证的。所以通常说它是一种不可靠的传输层协议，但事实上UDP协议的数据通信也是比较可靠的，只是相对TCP来说不那么可靠而已，因为它可以通过应用层协议的可靠性来保证数据传输的可靠性。

总体来说，UDP具有以下几个方面的明显特性：

（1）无连接性。

UDP可以提供无连接的数据报服务，这也决定了在使用UDP进行数据传输前是不需要建立专门的传输连接的，当然在数据发送结束时也无需释放连接。

（2）不可靠性。

因为UDP传输数据时是不需要事先建立专门的传输连接的，所以它的传输是不可靠的（但会尽最大努力进行交付），很可能传输不成功，特别适用于一些短小消息类的数据传输，如DHCP、DNS中的一些消息就是采用UDP协议进行传输的。

（3）以报文为边界。

UDP协议是直接对应用层提交的报文进行封装、传输，不拆分，也不合并，保留原来报文的边界，所以UDP是报文流，而前面介绍的TCP协议是字节流。因为UDP协议不拆分报文，自然也就没有报文段之说，但UDP报文传输到网络层后，在网络层仍然可以根据网络的MTU值进行分割。

（4）无流量和拥塞控制功能。

使用UDP协议进行数据传输时不能进行流量控制和拥塞控制，因为这类数据传输的连续性要比数据的完整性更重要，允许数据在传输过程中有部分丢失，如IP电话、流媒体通信等。

（5）支持各种交互通信方式。

在本章前面介绍了，TCP不支持组播、广播通信方式，只支持一对一的单播方式，但UDP支持各种通信方式，可以是一对一、一对多、多对一和多对多，所以凡是广播、组播通信方式都是采用UDP协议进行传输的。

计算机网络中有许多使用UDP协议的应用服务，如DNS、SNMP、DHCP、RIP、OSPF等。

10.7.2 UDP数据报头部格式

UDP协议数据报格式比较简单，如图10-56所示，其中“数据”字段是来自应用层的数据内容，其他部分为UDP协议头，总共8个字节。

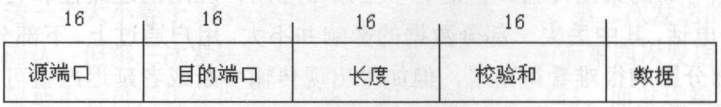


图10-56 UDP协议数据报格式

UDP 协议头各字段的说明如下：

- 源端口：标识源计算机上使用的 UDP 端口，占 16 位（2 个字节）。但这个字段是可选的，仅当需要目的主机返回一个应答时才有意义。如果不使用它，则此字段值为 0。
- 目的端口：标识目的计算机上使用的 UDP 端口，占 16 位（2 个字节）。如果目的主机应用层没有对应端口的应用进程，则该 UDP 数据报会被丢弃。
- 长度：标识此 UDP 数据报的总长度（包括 UDP 协议头和“数据”部分），以字节为单位，占 16 位（2 个字节）。因为一个 IP 数据报的最大长度为 65535 个字节（64KB），所以理论上，UDP 数据报“数据”部分的最大长度为 65535-20（最小 IP 报头）-8（UDP）=65507 个字节。以上只是针对网络层 IP 协议数据报大小的限制，再向数据链路层传输时同样要受到对应数据链路协议帧中“数据”字段大小（MTU）的限制，如以太网中的“数据”字段最大不能超过 1500 个字节，这样一来，UDP 报文要在网络层向以太网链路传输时不被分片，则其最大大小为 1500-20-8=1472 个字节。当然这只是在以太局域网中，在互联网中由于各段链路的 MTU 值不一样，为了不被频繁分片，其长度通常限制在 512 个字节。



至于为什么是 512 个字节，笔者在这里给大家分析一下，因为目前也查不到任何权威的答案。我想这是因为 TCP 协议在 Internet 上进行数据传输时默认采用的 MTU 值为 576 个字节，减去 IPv4 协议头部最小为 20 个字节，再减去 UDP 协议头部的 8 个字节，这样 UDP 报文中“数据”字段的最大值理应为 548 个字节。但为了能满足大多数情况，在 UDP 协议的 MTU 取值上考虑到了 IP 协议头部有可能有“选项”字段，如果取“选项”字段的最大值，则 UDP 协议的 MTU 值为 508 个字节。至于为什么是 512 个节字，我是这么理解的，因为 508 不等于 2^n，取其最近的 2^9，则正好等于 512 个字节。因为事实上 IP 协议头部包括了 40 个字节“选项”字段的话，则已使 IPv4 头部长度最大了，所以取 512 个字节也完全满足了绝大多数的 IP 网络通信需求。

- 校验和

在进行校验和计算时，需要在 UDP 数据报头部前面加上 12 个字节的伪头部（如图 10-57 所示，它不是数据中的真实成分，也不用来传输，仅用来计算校验和），然后对整个 UDP 头部和“数据”部分进行校验。通常是采用 CRC 校验方式，具体参见 8.3.4 节的相关内容。

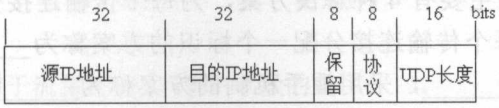


图 10-57 UDP 伪头部

UDP 协议的伪头部仅用于校验，目的是让数据报接收者确定发送、接收的 UDP 数据报是来自正确的源且是发给自己的，既检验了 IP 数据报的源 IP 地址和目的地址，又保证了 UDP 数据单元

到达正确的目的地址，因为 UDP 结构中只包含了源 UDP 端口和目的 UDP 端口，没有 IP 地址信息。

其中的“源 IP 地址”和“目的 IP 地址”字段与 IPv4 头部的对应字段一样，“保留”字段中的 8 位全取 0，“协议”字段就是 UDP 协议编号，为十六进制的 0x17，“UDP 长度”字段为整个 UDP 报文长度，不包括伪头部。

因为 UDP 协议无连接管理，无差错控制，也无拥塞控制，所以比较简单，再加上在下一章介绍一些具体网络应用时会对 UDP 协议功能有更全面的体现，所以本章仅介绍了这些内容。

10.8 课后自测题

为了巩固本章所学的内容，请认真做下面的自测题，参考答案见附录。

10.8.1 填空题

1．传输层服务也有两大类，其中_____可提供流量控制、差错控制和序列控制等功能，代表协议是_____，而_____只能提供不可靠的服务，代表协议是_____。

2．传输层提供的服务可分为_____和_____两种。传输层所提供的服务等级可分为 3 类，传输差错率和故障率比较低的为_____类，传输差错率和故障率比较高的为_____类，传输差错率和故障率居中的为_____类。

3．在传输层的 TPDU 中用于传输连接建立阶段的有_____和_____，用于数据传输阶段的有_____、_____、_____和_____，用于传输连接释放阶段的有_____和_____。

4．5 种基本的传输服务原语是_____、_____、_____、_____和_____。在 ISO 网络中，各种传输服务原语又可分为_____、_____、_____和_____四大类。

5．ISO 网络主要有 10 种传输服务原语，其中用于传输连接建立阶段的有_____、_____、_____和_____，用于数据传输阶段的有_____、_____、_____和_____，用于传输连接释放阶段的有_____和_____。

6．传输层的寻址方案有两种，其中_____用于为非常规应用层服务提供传输层寻址，是使用_____来监听这些应用进程的；_____用于为常规应用层服务提供传输层寻址，是使用_____来监听这些应用进程的。

7．避免重复传输连接主要有 4 种解决方案，为每个传输连接分配一个新的 TSAP 地址的方案称为_____，为每个传输连接分配一个标识的方案称为_____，定义和计数每个分组的生存时间的方案称为_____，采用握手机制的方案称为_____。

8．套接字原语中，应用服务器主要调用的有_____、_____、_____和_____这 4 个，应用服务客户端主要调用的有_____、_____、_____和_____这 4 个。

9．在 TCP 数据传输中，用来确保传输可靠性的机制包括_____、_____、_____和_____这 4 种。

10. TCP 的拥塞控制机制主要包括_____、_____、_____和_____这 4 种。

10.8.2 选择题（可多选）

1. 在端到端数据传输系统中最高层是（ ）

A. 网络层　　B. 传输层　　C. 表示层　　D. 会话层

2. 在 OSI/RM 七层结构中既属于资源子网又面向应用的是（ ）。

A. 数据链路层　　B. 网络层　　C. 会话层

D. 表示层　　E. 传输层

3. 下列传输层服务类别中具有多路复用功能的是（ ）。

A. TP0　　B. TP1　　C. TP2

D. TP3　　E. TP4

4. 下列传输层服务类别中具有差错检测功能的是（ ）。

A. TP0　　B. TP1　　C. TP2

D. TP3　　E. TP4

5. 下面关于 TP4 和 TCP 协议的区别与联系说法不正确的是（ ）。

A. TP4 与 TCP 一样也是面连接的传输层协议类别

B. TP4 也支持无连接传输服务

C. TP4 是 ISO 网络中的传输层协议类别，TCP 是 TCP/IP 网络中的具体传输层协议

D. TCP 协议包括在 TP4 之中，TP4 同时还包括 UDP 协议

6. TPDU 头部是以下（ ）部分组成的。

A. 长度指示（LI）字段、固定部分、可变部分

B. 长度指示（LI）字段、数据字段、固定部分

C. 数据字段、固定部分、可变部分

D. 长度指示（LI）字段、数据字段、可变部分

7. 下列 TPDU 中不支持 TP0 类别传输层协议的是（ ）。

A. DR　　B. DC　　C. DT

D. AK　　E. ED

8. 下列 TPDU 中不可以携带有效载荷的是（ ）。

A. CC　　B. DC　　C. DT

D. AK　　E. ED

9. 下列 TPDU 中有用于流量控制的 CDT 字段的是（ ）。

A. CR　　B. CC　　C. DT

D. AK　　E. ED　　F. EA

10. 下列 TPDU 中需要编号的是（ ）。

A. CR　　B. CC　　C. DT

D．AK　　E．ED　　F．EA

11．下列 TPDU 中指明了希望接收的下一个 TPDU 序列号的是（　　）。

A．CR　　B．CC　　C．DT

D．AK　　E．ED　　F．EA

12．下列 TPDU 字段中相当于 TCP 数据传输中的“确认号”字段的是（　　）。

A．目的参考　　B．TPDU 序列号和 EOT

C．CDT　　D．你的 TPDU 序列号

13．下列 ISO 网络传输服务原语中属于“指示”类型的是（　　）。

A．T-CONNECT.Request　　B．T-CONNECT.indication

C．T-CONNECT.response　　D．T-CONNECT.confirm

E．T-DATA.indication　　F．T-DATA.request

14．每条 TCP 传输连接只能有两个端点，其中“端点”指的是（　　）。

A．通信双方主机或节点　　B．通信双方的 IP 地址

C．通信双方所用的传输端口　　D．通信双方所用的套接字

15．在 TCP 传输的数据段中，每个数据段的大小必须小于（　　）个字节。

A．65535　　B．1500　　C．65515　　D．1518

16．假设当前发送的一个 TCP 数据段大小为 100 个字节，“序号”字段值为 100，则下一个发送的 TCP 数据段中的“序号”字段值为（　　）。

A．101　　B．200　　C．199　　D．201

17．下面关于 TCP 数据段中的“确认号”说法正确的是（　　）。

A．“确认号”字段只在应答数据段才会赋值

B．“确认号”字段值代表本端已经正确接收到的最后一个数据字节的序号

C．“确认号”字段值代表本端期望接收到对方下一个数据段中的第一个字节序号

D．“确认号”字段值代表本端已经正确接收到该序号以前的所有数据

18．如果某 TCP 数据段的“数据偏移”字段值为 10（此处为十进制），则该数据段中的“数据”部分是从整个 TCP 数据段数据部分的第（　　）个字节开始的。

A．10　　B．20　　C．40　　D．41

19．假设本次所发送的数据段的“确认号”字段值为 501，而“窗口大小”字段值是 100，则表明希望对端下次发送的数据段字节序号范围为（　　）。

A．501～600　　B．500～599　　C．100～200　　D．101～199

20．当全部数据传输完后要释放一个 TCP 传输连接时，则要将所发送的数据段的（　　）字段置 1。

A．PSH　　B．SYN　　C．FIN

D．RST　　E．URG

21．下面关于 TSAP 和套接字关系的说法正确的是（　　）。

A．套接字就是 TSAP

B．TSAP 只是套接字中的传输层端口部分

C．TSAP 和套接字都位于传输层上边缘，作为与应用层交互的逻辑接口

D．一个 TSAP 对应一个套接字

22．在下列套接字原语中，应用服务器和应用客户端都可以调用的是（　）。

A．SOCKET　　B．BIND　　C．ACCEPT

D．CONNECT　　E．SEND　　F．CLOSE

23．每个 TCP 连接的最初和最终状态是（　）。

A．CLOSING　　B．CLOSE WAIT

C．CLOSED　　D．LISTEN

24．在一个 TCP 传输连接中，本端主机已收到对端关闭传输连接确认，等待对端发送关闭传输连接请求，则此时本端主机进入（　）状态。

A．TIMED WAIT　　B．FIN WAIT 1

C．FIN WAIT 2　　D．CLOSE WAIT

25．在 TCP 传输连接建立时，在服务器端创建套接字的过程中不需要调用（　）套接字原语。

A．CLOSED　　B．SOCKET　　C．BIND

D．LISTEN　　E．ACCEPT

26．假设每个数据段大小为 100 个字节，现某主机接收到了序号为 1、101、201、401 的 4 个数据段，过一段时间又收到了 301 号数据段，则此时该主机向对方发送的确认数据段中的“确认号”字段值为（　）。

A．501　　B．500　　C．301　　D．300

27．假设每个数据段大小为 100 个字节，现某主机先后收到了序号为 1、201、101、301、601、501、801 的 7 个数据段，则此时该主机可向对方发送的确认数据段中的“确认号”字段值和可传输给应用层的数据段分别是（　）。

A．901 已接收到的全部数据段　　B．701　1、201、101、301、601、501

C．101　1　　D．401　1、201、101、301

28．在“慢启动”拥塞解决方案中，当发生一次数据丢失时，现假设前一次数据传输时的 CWND 为 32MSS 个字节，则此时的 SSTHRESH 和 CWND 分别为（　）。

A．32MSS　16MSS　　B．16MSS　1MSS

C．16MSS　2MSS　　D．32MSS　1MSS

29．在 TCP 的数据传输过程中第三次发生数据丢失时，将进入（　）数据发送过程。

A．慢启动　　B．拥塞避免　　C．快速重传　　D．快速恢复

30．下面关于 UDP 协议的说法不正确的是（　）。

A．是一个无连接的传输层协议

B．是一个不可靠的传输层协议

C．使用 UDP 传输数据时，不需要服务器监听，也不需要客户端建立传输连接

D．可以进行流量控制，同时支持单播、多播和广播传输方式

31．下列关于 TCP 和 UDP 协议说法错误的是（　　）。

A．TCP 是可靠的传输层协议，UDP 不可靠

B．TCP 只支持单播传输，UDP 支持各种传输方式

C．TCP 是面向连接的，而 UDP 是无连接的

D．TCP 数据段下发到数据链路层时可以进行分段，UDP 报文下发到数据链路层时不能分段

32．每个 UDP 数据报“数据”字段的大小必须小于（　　）个字节。

A．65535　　B．1500　　C．65515　　D．65507

33．在以太网中，TCP MSS 选项最大可以设置（　　）。

A．1500　　B．1518　　C．1480　　D．1460

34．下面关于 MSS 选项说法错误的是（　　）。

A．指定本端可接收的一个 TCP 数据段大小

B．指定本端可接收的一个 TCP 数据段中“数据”字段的大小

C．MSS 选项仅在 SYN 数据段中存在

D．MSS 的最大设置=网络层 MTU-网络层协议头-TCP 协议头

35．下面关于 TCP 数据段确认说法正确的是（　　）。

A．可以一次性对多个数据段进行确认

B．仅对收到的连续数据段进行确认

C．“确认号”代表对方已正确接收的最大数据段序号

D．接收到的不连续数据段也可以先向应用层上传

36．关于 TCP 和 UDP 端口，下列说法中正确的是（　　）。

A．TCP 和 UDP 分别拥有自己的端口号，二者互不干扰，可以共存于同一台主机

B．TCP 和 UDP 分别拥有自己的端口号，但二者不能共存于同一台主机

C．TCP 和 UDP 的端口号没有本质区别，二者互不干扰，可以共存于同一台主机

D．TCP 和 UDP 的端口号没有本质区别，但二者互相干扰，不能共存于同一台主机

37．UDP 协议的“校验和”字段所校验的数据是（　　）。

A．UDP 头部　　B．整个 UDP 报文

C．UDP 头部和伪头部　　D．UDP 伪头部和“数据”部分

38．TCP 头部的最小和最大长度分别是（　　）个字节。

A．20　40　　B．20　60

C．18　40　　D．40　60

39．在图 10-58 所示的 TCP 连接的建立过程中，SYN 中的 Z 部分应该填入（　　）。

A．a　　B．b

C．a+1　　D．b+1

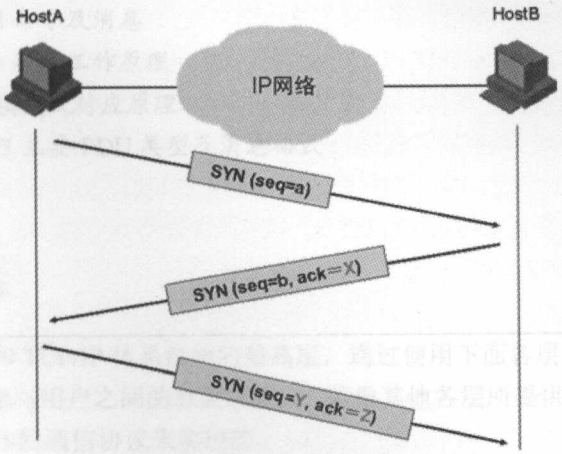


图 10-58 TCP 连接的建立过程

40. 在图 10-59 所示的 TCP 连接的拆除过程中，FIN 中的 Y 部分应该填入（ ）。

A. p B. q C. p+1 D. q+1

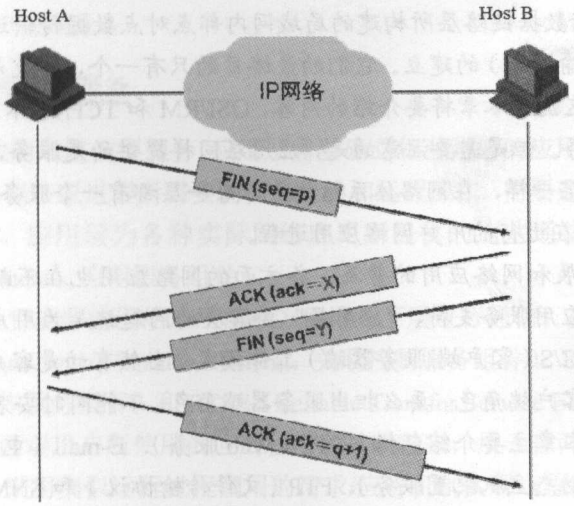


图 10-59 TCP 连接的拆除过程

11 应用层

前面各章主要完成了两方面的功能：一是网络连接的建立（或者说是网络通信平台的搭建）；二是数据传输通道（包括数据链路层所构建的局域网内部点对点数据传输通道和传输层所构建的不同网络间端对端数据传输通道）的建立。它们的最终目的只有一个，那就是为我们所需进行的网络应用搭建有效的平台，这就是本章将要介绍的内容，OSI/RM 和 TCP/IP 体系结构的最高层——“应用层”所提供的应用服务。但是需要注意的是，应用层同样提供的是服务，而不是提供具体的网络应用软件。但与其他各层一样，在网络体系结构中，每一层都有一个服务作用实体，“应用层”的实体（AE）就是各种正在进行的用户网络应用进程。

随着网络技术的发展和网络应用的普及，各方面的网络应用也在不断涌现，非常繁多。每类网络应用都需要对应的应用服务支持，“应用层”要解决的问题就是为用户提供所需的应用服务。这些网络应用服务都是 C/S（客户端/服务器端）工作模式，当然有的是客户端与服务器端分离（也就是一台主机要么担当客户端角色，要么担当服务器端角色，不能同时安装两种角色的服务软件），有的是两者同时支持。本章主要介绍包括 HTTP（Web 服务）、E-mail（电子邮件服务）、DNS（域名解析服务）、DHCP（动态主机配置服务）、FTP（文件传输协议）和 SNMP（简单网络管理协议）这 6 类应用服务的基本工作原理。

本章重点与难点：

- HTTP 请求/响应报文格式
- DNS 名称空间和名称服务器分类
- DNS 报文格式以及递归、迭代解析原理
- DHCP 报文分类及格式
- DHCP 服务 IP 地址自动分配原理

- SMTP 和 POP3 命令及消息
- SMTP 和 POP3 服务工作原理
- FTP 两种工作模式及对应原理
- SNMPv1/v2c/v3 主要 PDU 类型及消息格式

11.1 应用层概述

应用层是 OSI/RM 和 TCP/IP 体系结构的最高层，通过使用下面各层所提供的服务直接向用户提供服务，是计算机网络与用户之间的界面或接口。就像其他各层所提供的服务一样，应用层的各种服务功能也是通过具体的通信协议来实现的。

应用层的网络应用服务非常多，而且有许多是我们天天在用的，如 Web 服务（http）、电子邮件服务（POP3、SMTP 等）、DNS 域名服务、DHCP IP 地址自动分配服务、FT0050 和 TFTP 文件传输服务、Telnet 和 STelnet 远程登录服务等，各种动态路由协议（RIP、OSPF、IS-IS 等）也是应用层的服务。随着互联网应用的普及和发展，各种新的网络应用服务更是层出不穷，本章仅介绍在 TCP/IP 网络中最常用的一些服务。

11.1.1 应用层组件及典型应用服务

为了向用户提供有效的网络应用服务，应用层需要确立相互通信的应用程序或进程的有效性并提供同步，需要提供应用程序或进程所需要的信息交换和远程操作，需要建立错误恢复的机制以保证应用层数据的一致性。应用层为各种实际网络应用所提供的这些通信支持服务统称为 ASE（Application Service Element，应用服务组件）。

不同的 ASE 使得各种实际的网络应用能够方便地与下层进行通信。其中最重要的 3 个 ASE 分别是 ACSE（Association Control Service Element，关联控制服务组件）、ROSE（Remote Operation Service Element，远程操作服务组件）和 RTSE（Reliable Transfer Service Element，可靠传输服务组件）。ACSE 可以将通信两端用户所使用的应用程序名关联起来，用于在两端应用程序之间建立、维护和终止连接；ROSE 采用类似远程过程调用的请求/应答机制实现远程操作；RTSE 则通过先将数据转化为八位字节串，然后将数据串分解为多个段，最后将每段传送到表示层，以便发送在分段之间建有检验点。通过表示层服务，RTSE 使用会话层活动管理服务来管理数据分段的传输。

除了以上提到的 3 个应用服务组件外，OSI/RM 的应用层提供了 5 种不同的网络应用类型来实现不同的应用需求，它们是 MHS（Message Handling System，报文处理系统）、FTAM（File Transfer, Access and Management，文件传输、存取和管理）、VTP（Virtual Terminal Protocol，虚拟终端协议）、DS（Directory Service，目录服务）、TP（Transaction Processing，事物处理）、RDA（Remote Database Access，远程数据库访问）。但由于目前 OSI/RM 只是起到参考模型的作用，所以并没有实际的网络应用是按照上述协议实现的。而在 TCP/IP 协议体系结构的应用层却相反，拥有许多主流的应用

层协议和基于这些协议实现的 TCP/IP 应用。

TCP/IP 协议体系结构中的应用层解决了 TCP/IP 网络应用所存在的共性问题，包括与网络应用相关的支撑协议和应用协议两大部分。其中的支撑协议包括域名服务系统（DNS）、动态主机配置协议（DHCP）、简单网络管理协议（SNMP）等；典型的应用服务包括 Web 浏览服务、E-mail 服务、文件传输访问服务、远程登录服务等，与以上这些典型网络应用服务相关的协议包括超文本传输协议（HTTP）、简单邮件传输协议（SMTP）、文件传输协议（FTP）、简单文件传输协议（TFTP）和远程登录（Telnet）等。

11.1.2 应用层 C/S 服务模型

网络应用软件之间最常用、最重要的交互模型是 C/S（Client/Server，客户端/服务器）模型。如 Web、E-mail、DNS、DHCP、FTP/TFTP、Telnet 等都是用这种模型工作的。这里所说的“客户端”和“服务器”分别是指通信双方主机上所安装的对应应用程序。

网络上的应用程序之间为了能顺利地进行通信，一方通常需要处于守候状态，等待另一方请求的到来。这就是本节所介绍的 C/S 模型，首先服务器需要时刻守候各种应用服务（通常是通过监听各个传输端口来实现的），然后客户端向服务器发出对应的应用会话请求，最后服务器对客户端的会话请求作出响应。如图 11-1 所示是 TCP/IP 协议体系结构中一个通过互联网进行交互的 C/S 模型示例。在图中，服务器处于守候状态并监听客户端的会话请求。在客户端发出应用会话请求命令并经过互联网传输给服务器后，服务器接收到这个会话请求并执行相应会话请求所指定的任务，最后将执行的结果以应答消息经互联网返回送给客户端。

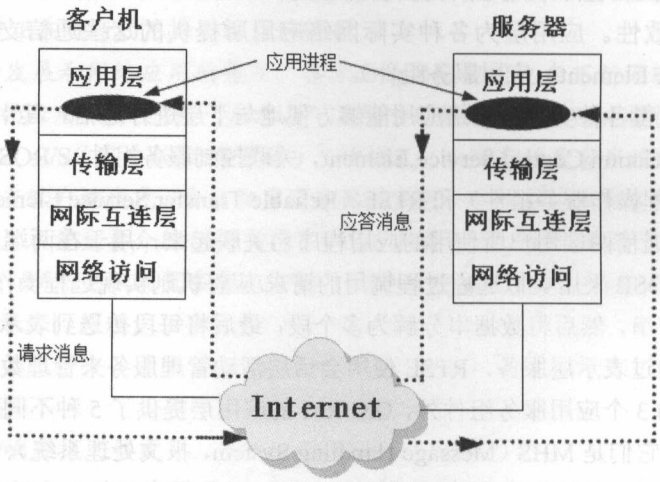


图 11-1 应用服务的 C/S 模型示例

一台主机上通常会同时运行多个应用服务器程序，每个服务器程序需要并发地处理多个客户的请求，并将处理的结果返回给客户端。因此，服务器程序通常比较复杂，对主机的硬件资源（如

CPU 的处理速度、内存的大小等）及软件资源（如分时、多线程网络操作系统等）都有一定的要求。而客户程序由于功能相对简单，通常不需要特殊的硬件和高级的网络操作系统。

C/S 模型不但很好地解决了互联网应用程序之间的同步问题（何时开始通信、何时发送信息、何时接收信息等），而且 C/S 模型的这种非对等相互作用的特点很好地适应了互联网资源分配不均的客观事实，因此成为互联网应用程序相互作用的主要模型。

在 TCP/IP 互连网络中，服务器程序通常使用 TCP 协议或 UDP 协议的端口号作为自己的特定标志。在服务器程序启动时，首先在本地主机注册自己使用的 TCP 或 UDP 端口号，这样服务器程序在声明该端口号已被占用的同时，也通知本地主机如果在该端口上收到信息，则需要将这些信息转交给注册该端口的服务器程序处理。在客户程序需要访问某个服务时，可以通过与服务器程序使用的 TCP 端口建立连接（或直接向服务器程序使用的 UDP 端口发送信息）来实现。

在互连网络中，客户发起请求完全是随机的，可能出现多个请求同时到达服务器的情况，因此服务器必须具备处理多个并发请求的能力。为了应对客户端的并发多请求情形，服务器的部署可以采用以下两种方案之一：

（1）迭代服务器（Iterative Server）方案。

在迭代服务器解决方案中，服务器程序中包含一个请求队列，客户请求到达后先进入队列中等待，服务器按照先进先出的原则对这些客户端请求一个个作出响应。只有前一个请求处理完后服务器才会处理下一个请求。

（2）并发服务器（Concurrent Server）方案。

并发服务器是一个守护进程，在没有请求到达时它处于等待状态。一旦客户请求到达，服务器立即再为之创建一个子进程，然后回到等待状态，由子进程响应请求。当下一个子进程到达时，服务器再为之创建一个子进程。其中，并发服务器叫做主服务器，子进程叫做从服务器。

11.2 Web 服务基础

如果说现在哪种网络应用最为普及，应用频次最高，相信用过计算机的人都会很自然地想到我们天天都在进行的“网站访问”，其实也就是我们通常所说的万维网（World Wide Web）服务，简称 WWW 服务或者 Web 服务。Web 服务的核心应用层协议就是 HTTP（Hypertext Transfer Protocol，超文本传输协议），可以把 HTTP 看成是 Web 服务的“幕后英雄”。

我们进行网站访问时好像觉得非常简单，点一下鼠标或者输入一个网站名称就进去了，其实它仍需要一整套 Web 服务功能来支持，其中涉及到许多协议和网络应用程序功能。下面先来分析一下 Web 服务模型或者 Web 服务系统结构。

11.2.1 Web 服务模型

任何网络应用服务都不是单一程序或者单一命令可以实现的，它必须有一个能实现服务请求、服务提供和服务注册这 3 个方面的完整系统结构，这就是我们通常所说的“服务模型”。

从用户角度来看，Web 服务或者所有互联网网站其实就是一个全球范围内的巨大文档，更具体地说是 Web 页面的集合。有了以上基本认识就可以很容易得出 Web 服务模型，如图 11-2 所示。它包括 3 个部分：Web 服务提供者（Web 服务器）、Web 服务请求者（Web 客户）和 Web 服务注册中心（互联网注册、管理机构），其中“Web 服务注册中心”是可选项，仅在互联网中 Web 服务才是需要注册的，在局域网中是不需要的。

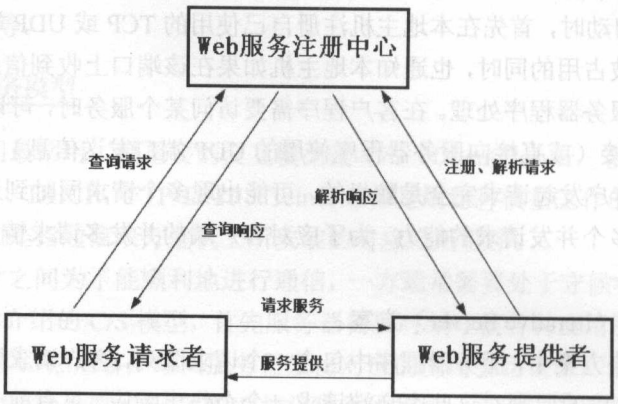


图 11-2　Web 服务模型

下面简单介绍这 3 个部分的基本功能。

- Web 服务注册中心：这是一个可选部分，指互联网注册、管理中心，并提供网站域名解析的 ISP（Internet 服务提供商），在仅用于局域网内部的网站访问中无需此部分。它负责互联网网站域名的注册和管理，并向用户提供互联网域名解析服务。
- Web 服务提供者：是指 Web 服务器或者直接叫做 Web 网站，是 HTTP 服务器端。它是由一些 Web 服务器程序（如 IIS、Apache 等）开发的，除了要对网站本身进行描述和向注册中心注册外，更重要的职责是为 Web 访问用户提供所需的网页信息，还可能包括授权用户账户系统的创建、授权和维护。
- Web 服务请求者：是指 Web 用户在访问网站时所用的 HTTP 客户端程序，如各种浏览器（如 IE、Chrome、360 浏览器等）。在 Web 浏览器中输入网站的域名或 IP 地址或者在其他网站或文档上点击该网站的链接即可进入到对应的网站，然后再点击相应的页面链接访问到所需要的文字、图片、音/视频等页面。当然如果要访问受限的 Web 服务器，则还需要向 Web 服务提供合法的用户账户信息。

以上各部分均包括一些协议，并由支持这些协议的网络应用软件来实现。本节仅介绍“Web 服务提供者”和“Web 服务请求者”这两部分的功能实现原理。下面先了解一下万维网的一些基本特性。

11.2.2　万维网的全球统一标识

要在浩瀚的互联网大海中访问到你需要访问的那一个网站或一个网页，如果没有一个基于全球

范围的唯一标识是无法想象的。在万维网中存在 3 种全球统一标识，即 URL（Uniform Resoure Locator，统一资源定位器）、URI（Uniform Resource Identifier，统一资源标识符）和 URN（Uniform Resource Name，统一资源名称）。

这 3 种标识不仅看起来非常相似，而且关系非常密切。从它们的中文名称可以看出，URL 用来定位资源，标识的是访问对应资源的路径；URI 是用来标识资源信息的，而 URN 则是用来为资源命名的。但是 URI 中的标识符可以是资源的名称，也可以是资源的地址。另外，URI 在 RFC1630 中被定义，发布于 1994 年 6 月；URL 在 RFC1738 中被定义，发布于 1994 年 12 月；URN 在 RFC1737 中被定义，发布于 1994 年 12 月。从中也可以看出，是先有的 URI，然后才有的 URL 和 URN，URL 和 URN 是 URI 扩展后的子集。

1. URL

在互联网上的网站，无论它的服务器位于哪里，你用家里连接上互联网的计算机，在浏览器中输入这个网站的网址就可以轻易地访问到该网站或者该网站上的对应页面。这就是 Web 网站上所有资源的统一定位标识特性——URL，是后面将要具体介绍的 URI 中的一种。

URL 用于标识一个互联网资源并指定对其进行操作或取得该资源的方法。这里所说的“资源”是指在互联网上可以被访问的任何对象，包括文件目录、文件、文档、图片、图像、音/视频，甚至是电子邮件地址等。

URL 采用全球统一的格式来标识万维网上资源的访问地址，以确保每个互联网网站和网站页面在全球范围内都有唯一的地址标识。URL 的一般格式（其实后面还可以有可选项）为：

```
<协议>://<主机名>:<端口>/<路径>
```

（1）协议。

这里的“协议”参数是指访问对象所使用的协议，包括 HTTP、FTP、GOPHER、TELNET、FILE、MAILTO 等。在协议后必须加上“://”。

（2）主机名。

“主机名”参数是指 Web 服务器名称，互联网上的服务器主机名就是网站的域名，用来指定你要访问的资源在哪台服务器主机上。互联网的主机名通常是以 WWW 开始的，后面必须跟着“:”（冒号），以指定访问对象建立通信连接时所用的传输层端口。

（3）端口。

“端口”参数是指定在建立连接时所用的传输层端口号，如果采用的是常规协议，端口号可以省略，直接使用默认的端口号。如 HTTP 协议的默认端口号为 80，FTP 协议的默认端口号为 21，TELNET 协议的默认端口号为 23。即使是常规协议，所使用的端口号也可以改，但一定得与 Web 服务器上所设置的端口一致，否则访问不了。后面跟的“/”代表服务器的根目录。

（4）路径。

“路径”参数是指定要访问的对象在 Web 服务器上的文件路径，与本地主机上的文件路径格式一样，以根目录（/）开始。如果要访问的是网站首页，则不用输入此部分，直接以前面的“/”符号访问服务器的根目录。

例如我们通常要访问 51CTO 网站，则可以在浏览器的地址栏中输入 http://www.51cto.com，当然现在的浏览器我们可以不用输入“http://”和“/”这两部分了，浏览器会自动为我们加上的，没有指定端口，直接使用 HTTP 协议使用的默认 80 号端口。

如要直接访问我在 51CTO 博客中的一篇文章，在浏览器的地址栏中输入 http://winda.blog.51cto.com/55153/868663，winda.blog.51cto.com 部分是我在 51CTO 博客上的博客域名，后面的/55153/868663 就是这篇文章的路径。

如要访问一个域名为 ftp.winclass.net 的 FTP 站点，则可以表示为 ftp://ftp.winclass.net，当然这里是没有考虑访问用户账户的；如要访问一个 winda@wincalss.net 邮箱，则可以输入 mailto:winda@wincalss.net。

2. URI 和 URN

前面说了，URL（统一资源定位器）和 URN（统一资源名称）都是 URI（统一资源标识符）的子集，可被视为 URL、URN 或两者兼备。URL 我们前面已有介绍，URN 仅是指资源的名称，如国际书号、互联网邮箱账户、RFC 文档等。URN 如同一个人的名称，而 URL 代表一个人的住址，但这两种统一标识实际用的都不多，现在主要用的还是 URL。

URI 的基本语法格式如下：

```
<方案名>:<资源标识>/<资源路径>
```

这里“方案名”代表的是对资源进行标识的协议（但有时不是直接用协议名本身显示），“资源标识”是指保存资源的服务器域名，“资源路径”是资源的保存路径，它不会因为访问路径变化而变化。几种常见类型 URI 的基本格式如下：

- mailto:*mbox@domain*：指定通过 SMTP 协议发送邮件的邮箱地址。
- ftp://*user:pass@server/file*：指定要通过 FTP 协议传输文件的用户账户和文件位置。
- http://*domain/path*：指定要通过 HTTP 协议访问的网站或网页的路径。
- file://*host/file*：标识在本地磁盘或者网络文件系统上的文件地址。

其中第一个既可以说是一个 URN，也可以说是一个 URL，标识的是一个邮箱名称，其他 3 个又是 URL。

11.2.3 万维网文档标记

因为 Web 用户所使用的操作系统、浏览器类型多种多样，它们所使用的开发语言也可能完全不同，所以为了使不同的用户计算机上都能正确地显示万维网上的网页信息，也必须为网站的信息制定全球统一的表示方法（称为“标记”方法），那就是本节要介绍的重点 HTML（HyperText Markup Language，超文本标记语言）。

HTML 是一种万维网的标记语言（不是一种开发语言），用来结构化信息如标题、段落和列表等，也可用来在一定程度上描述文档的外观和语义，由万维网协会（World Wide Web Consortium，W3C）维护。使用 HTML 开发的网页文件扩展名为.html（早期 DOS 时代采用的是三位扩展名，

所以那时的扩展名为.htm)。目前主流使用的是 HTML 4.0 版本，但现在 HTML 5.0 版本也已经发布，并即将全面进入实用阶段。

HTML 使用一套标签(tag)来标记网页内容的格式、排版。HTML 标记标签通常被称为 HTML 标签（HTML Tag)。HTML 标签是由尖括号（<>）包围的关键词（关键字不区分大小写，但通常是以小写表示)，比如 <html>。而且 HTML 标签通常是成对出现的（但不是所有标签都是成对的，如换行标签
和分段标签<p>就只有一个标签)，后面一个标签前面要加上“/”符号，比如 和，第一个标签是开始标签，第二个标签是结束标签，开始标签和结束标签也被称为开放标签和闭合标签。如表示后面的字体采用斜体排版，表示此次采用斜体排版结束。常见的 HTML 标签如表 11-1 所示。详细的 HTML 知识请参考相关书籍或资料。

表 11-1 常见的 HTML 标签

标签	含义说明
标记网页结构类标签	
<html> </html>	中间部分为网页内容
<head> </head>	中间部分为网页头部
<title> </title>	中间部分为标题内容
<body> </body>	中间部分为网页正文
版面排版类标签	
<h#> </h#>	定义中间部分内容标题字号
 	定义中间部分内容字体大小
 	定义中间部分内容字型
 	定义中间部分内容文字颜色
	定义中间部分内容为小字体
<big> </big>	定义中间部分内容为大字体
	定义中间部分内容为粗体字
	定义中间部分内容为斜体字
<u> </u>	定义中间部分内容为下划线字
<strike></strike>	定义中间部分内容为删除线字
	定义中间部分内容为下标字
	定义中间部分内容为上标字
 	从此处换行
<p>	从此处分段

11.2.4 HTML 文档类型

上面讨论的 HTML 标记类文档在磁盘中保存时是以.html 为文件扩展名的。保存后，文档的内容是不会改变的，Web 每次打开这个文档所见到的内容都是完全一样的，除非经过了人为编辑、修改。这就是我们通常所说的“静态 HTML 文档”的概念。

静态 HTML 文档中，网页内容是静态分配的，用户每次访问时所看到的内容是一样的。但这里要区分两个概念，那就是“静态 HTML 文档”和“静态 HTML 元素”，这两者不是等同的概念，也没有一一对应的关系。也就是说，在静态 HTML 文档中可以有像文本、图片之类的静态网页元素，但现在的 HTML 4.0 及更新版本还可标记像图像、动画、音/视频这类的动态网页元素。

与“静态 HTML 文档”相对的就是“动态 HTML 文档”。动态 HTML 文档不是由管理员事先制作好的网页，而是即时由服务器应用程序或者脚本程序根据当前应用和用户提交的表单数据自动生成的，其显示的内容是在不断变化的。我们都应该有过网络购物的经历，我们在购物时，每次我们在提交定单时都会自动生成一个页面，其中显示了我们当次购物的商品、金额，以及收货地址等信息。尽管整个网页结构每次都是一样的，但网页中的内容可以说是每次都不一样，至少是购物商品、金额之类的栏目内容。这就是动态 HTML 文档。

既然动态 HTML 文档不是事先制作并保存在服务器磁盘上的，那么就需要有专门的网络应用程序来生成。当用户在网站上点击选择了某类应用时，Web 服务器就会调用对应的应用程序，然后把控制权交由该应用程序。而该应用程序接受用户在表单中输入的数据，用户确认提交表单后应用程序就会对所提交的数据根据对应的网络应用程序功能进行处理，然后在用户浏览器中输出处理的结果。由于用户每次提交的数据或者应用环境不一样，所以可能每次最终从 Web 服务器端返回的页面内容都是不一样的，就像我们每次在网站上购物所得的结算单都不一样，我们从铁道部网站查询车票时，不同时间得到的结果也是不一样的，因为每时每刻可售票数都在变化。

通过上面的分析我们可以看出，动态文档的创建至少需要满足两个条件：一是在 Web 服务器端要有能自动生成动态 HTML 文档的网络应用程序，这个我们可以不去了解，在进行 Web 服务器开发时程序设计师们会做的；我们要了解的是第二个条件，那就是 Web 服务器要能将用户在表单中输入的数据发送到对应的网络应用程序，然后 Web 服务器又能正确地把应用程序处理的结果以 HTML 文档方式回显给用户，这就是一种逻辑接口——CGI（Common Gateway Interface，公共网关接口）的功能。

CGI 是一种基于浏览器的输入，并可以在 Web 服务器上运行应用程序（统称 CGI 程序）的方法。它允许 Web 服务器与后端程序及脚本进行通信，同时后端程序和脚本又可以接受用户的输入信息（主要是来自表单），最后还能以 HTML 页面作为用户进行表单提交后的响应。基本流程如图 11-3 所示，具体描述如下（注意其中的序号与步骤是对应的）：

①用户在网站上点击了某项应用（如点击购买一个商品）后相当于向 Web 服务器提交了一个表单。

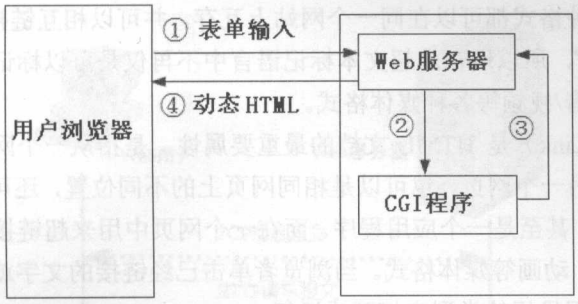


图 11-3 CGI 程序动态 HTML 生成基本流程

②Web 服务器在收到用户提交的表单后会通过在服务器程序设计中已设置好的关联功能把这个表单传送给对应的应用程序（也就是 CGI 程序）。

③CGI 程序接受用户提交的表单并对其进行处理（其中可能需要调用其他服务器或者客户主机，如各种数据库存储服务器），然后通过 CGI 程序的动态 HTML 生成功能把处理结果以一个自动生成的 HTML 返回到 Web 服务器。

④Web 服务器再把这个自动生成的 HTML 回显到 Web 用户的浏览器上。

CGI 程序其实都是脚本（Script）类程序。“脚本”通常是一些可执行命令（往往带有许多参数）的集合，但需要调用其他程序来解释或执行，不是计算机可直接处理的。这类解释或执行脚本的程序就是各种编程语言，特别是一些专门的脚本语言，如 Perl、JavaScript 和 Vbscript 等。

但是，CGI 并不是生成动态 HTML 的唯一方法，还有一种目前更广被采用的做法是在 HTML 页面中嵌入一些由编程语言编写的脚本（许多种编程语言都可以编写脚本，如 PHP、JSP、ASP 等，它们都可以接受并处理表单输入，同时与 Web 服务器进行交互），然后让 Web 服务器自己来执行这些脚本，以便生成最终的动态 HTML 发送给客户。这样一来，也就用 Web 服务器来调用 CGI 类应用程序了，直接由 Web 服务器来完成，效率更高。当然这仅适用于一些小的消息类动态 HTML 页面。

11.2.5 HTML 文档的“三超属性”

在万维网上有 3 个“超”字非常有吸引力，分别是“超文本”“超媒体”和“超接链”这 3 种属性。我们估且把它们称为万维网中的“三超属性”。

“超文本”（HyperText）其实也是一种文本格式，只是它可以用下面将要介绍的“超链接”的方法将保存在不同位置的文档或者文本信息组织在一起，构成一个相互关联的网状文本。如我们在访问网站时，经常见到有一些文字呈现不同颜色（一般为蓝色），用鼠标点击它后便弹出另外一个网页或者消息框，用于解释那部分制作链接的文字。

“超媒体”（HyperMedia）其实可以看成是前面“超文本”属性的扩展，把“超文本”中的文本媒体保存格式扩展到其他所有格式。早期的网站基本上是一些静态甚至纯文本的内容，现在的网站的内容就丰富多了，除了传统的文本信息、静态图片外，还有动态图像、动画、音/视频等各种

媒体格式。而且这么多种格式都可以在同一个网站上互存，并可以相互链接。但目前通常是把“超媒体”等同于“超文本”，所以现在的超文本标记语言中不再仅是可以标记文本格式，还可以标记像图片、图像、动画、音/视频等各种媒体格式。

“超链接”（HyperLink）是 HTML 文档的最重要属性，是指从一个网页指向一个目标的链接关系。这个目标可以是另一个网页，也可以是相同网页上的不同位置，还可以是一个图片、一个电子邮件地址、一个文件，甚至是一个应用程序。而在一个网页中用来超链接的对象可以是一段文本或者是一个图片、图像、动画等媒体格式。当浏览者单击已经链接的文字或图片后，链接目标将显示在浏览器上，并且根据目标的类型来打开或运行。

一个网站包括多级网页层次结构，如主页、次主页（主要是一些主要栏目）和普通网页，共同构成一个多层次的网站结构。我们访问网站时首先看到的是一个网站的主页，而在网站主页中又会把下一级的次主页以超链接的方式显示出来，这样用户就可以从网站主页上直接打开次主页，在次主页下面又可以有次次主页。因为一般对外公告的只是网站域名，亦即默认主页，如果在主页中没有把次主页的链接给出来，用户就不知道如何进到这些次主页中了。

当然每级主页下面又会有许多普通的网页，这些网页也会在对应级别的主页中显示出来，或者通过网页中的一些关键字超链接进入到具体的网页，毕竟再多的各级主页也不可能把全部的网页都以超链接的方式显示出来。通常是在各级主页中仅显示最新的普通网页超链接。

11.2.6 HTTP 服务访问基本流程

HTTP 是一个面向文本（text-oriented）的应用层协议，所使用的服务端口就是 TCP 80 端口（这就是 HTTP 服务的传输层地址），通信双方就是在这个端口上进行的。当然这个端口是可以更改的，但目前互联网上的 Web 服务器都是使用这个默认的 80 端口。每个 Web 服务器都有一个应用进程，时刻监听着 80 端口的用户访问请求。当有用户请求（以 HTTP 请求报文方式）来到时，它会尽可能地尽快做出响应（以 HTTP 响应报文方式），返回用户所访问的页面信息。当然，这一切都需要在传输层建立好对应的 TCP 连接基础之上。

至于何时关闭 TCP 连接，则要视所使用的 HTTP 协议版本而定，在 HTTP/1.0 及以前版本中，每一个请求－应答过程后就会关闭所使用的 TCP 连接，点击新的链接后又要重新建立新的 TCP 连接。这种 HTTP 服务方式称为非持续连接（No-persistent Connection）。而在 HTTP/1.1 版本后，HTTP 是允许在同一个 TCP 连接基础上对同一网站服务器上的多个不同页面进行访问的，仅当用户关闭对应的网站时对应的网站 TCP 传输连接才关闭，称为持续连接（Persistent Connection）。持续连接模式的 Web 服务访问基本流程如图 11-4 所示。

从图 11-4 可以看出，HTTP 服务的访问主要是 HTTP 消息的传递，其中包括 Web 用户向 Web 服务器提交的 HTTP 请求报文和 Web 服务器向 Web 用户返回的 HTTP 响应报文。下面先来综合了解一下 HTTP，然后再具体了解其报文结构。

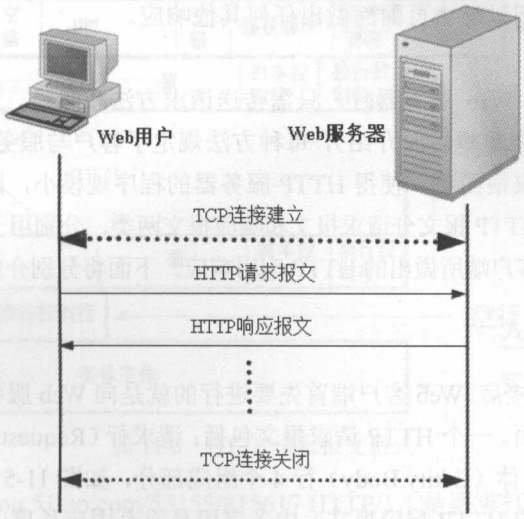


图 11-4 HTTP 服务访问的基本流程

11.2.7 HTTP 的主要特性

HTTP 是一个属于应用层的面向对象的协议，由万维网协会（W3C）和 Internet 工作小组（Internet Engineering Task Force，IETF）于 1990 年合作开发。在 1997 年前使用的是 RFC 1945 所定义的 HTTP/1.0 版本，1998 年后使用的是其更新版本 RFC 2616，即 HTTP/1.1 版本。HTTP 规定了 Web 客户端与 Web 服务器之间的报文交互方式，其报文中的每个字段都是一些 ASCII 字符串，而且各个字段的长度都是不固定的。

总体来说，HTTP 具有以下几方面的主要特性：

（1）客户/服务器（C/S）模式。

这是所有网络应用服务所采用的通用模式。Web 客户只需要使用支持 HTTP 的客户端程序（各种浏览器）即可访问由不同 Web 服务器程序开发的 Web 网站。

（2）无连接。

这里所说的“无连接”是指在进行 Web 应用前无需建立专门的 HTTP 应用层会话连接，仅需要直接利用传输层已为它建立好的 TCP 传输连接即可，而像 Telnet、SMTP、POP3 这类应用协议就是面向连接的了，除了需要传输层的 TCP 连接外，它们自己还要建立会话连接。

（3）高可靠性。

虽然 HTTP 本身是不可靠的无连接协议，但它使用了可靠的 TCP 传输层协议，在进行 HTTP 传输之前已建立了可靠的 TCP 连接，所以从数据传输的角度来讲，HTTP 的报文传输仍是可靠的。

（4）无状态。

这里所说的“无状态”是指同一客户第二次访问同一 Web 服务器上的同一页面时，服务器给客户的响应与第一次是一样的（当然，这是假设 Web 服务器上的对应页面没有更新），Web 服务器

不会记住这个客户曾经访问过这个页面而做出任何其他响应。

（5）简单快速。

客户通过 HTTP 访问 Web 服务器时，只需传送请求方法和路径。请求方法常用的有 GET、HEAD、POST 等（具体将在本章后面介绍），每种方法规定了客户与服务器之间进行的事务处理和消息类型。由于 HTTP 协议很简单，使得 HTTP 服务器的程序规模小，因而通信速度可以很快。

从图 11-4 可以看出，HTTP 报文分请求报文和响应报文两类，分别用于 Web 客户端发起的 HTTP 应用请求和 Web 服务器对客户端所做出的 HTTP 应用响应。下面将分别介绍这两种 HTTP 报文格式。

11.2.8 HTTP 请求报文格式

在建立好 TCP 传输连接后，Web 客户端首先要进行的就是向 Web 服务器发送 HTTP 请求报文，请求打开指定的网站或页面。一个 HTTP 请求报文包括：请求行（Request Line）、请求头部（Request Header）行、空行和实体主体（Entity Body）行 4 个组成部分，如图 11-5 所示。HTTP 报文（HTTP 请求报文和下节将要介绍的 HTTP 响应报文）中各字段是没有固定长度的。

1. 请求行

HTTP 请求报文中的“请求行”是由“请求方法”、URL 和“协议版本”这 3 个字段组成的，它们之间均以空格进行分隔。这部分是必不可少的。在请求行的最后有一个回车控制符和一个换行控制符（一起以 CRLF 表示），使下面的请求头部信息在下一行显示。

“请求方法”字段指示本请求报文中所使用的 HTTP 协议操作（其实就是指所使用的 HTTP 协议命令），具体如表 11-2 所示（注意全是大写字母，不能是小写的）。最常用的就是 GET 和 POST 这两种方法。

表 11-2 HTTP 请求方法

请求方法	含义说明
GET	请求服务器发送在 URL 字段中指定的 Web 页面。URL 的根部分是相对服务器的根目录，总以“/”前缀开始，下同
HEAD	请求读取 URL 字段指定的 Web 页面的头部信息，而不是全部的 Web 页面。利用这一方法可以得到一个页面的最后修改时间或者其他头部信息
PUT	请求在 URL 路径下存储一个 Web 页面，与 GET 方法相反
POST	在 URL 所指定的 Web 服务器后面附加一个以 URI 格式命名的资源，以便可以为 Web 服务器提供更多的信息，如在服务器上张贴一条海报消息
DELETE	删除 URL 字段指定的 Web 页面，当然最终能否删除成功还会受到用户权限的限制
TRACE	指明这是一个用来进行环回测试的请求报文，用于调试
CONNECT	用于连接代理服务器
OPTIONS	请求查询一些特定的选项信息

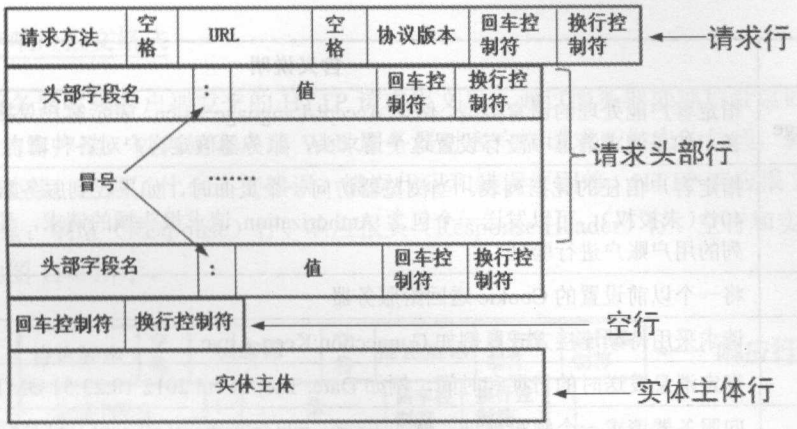


图 11-5 HTTP 请求报文格式

如 GET http://winda.blog.51cto.com/55153/815617 HTTP/1.1 是请求打开我在 51CTO 上博客的一篇文章，GET 是请求方法，中间部分是 URL，HTTP/1.1 是使用的 HTTP 协议版本，且它们三者之间用空格分隔。

2. 请求头部行

HTTP 请求头部行包括一系列的“请求头”和它们所对应的值，指出允许客户端向服务器端传递请求的附加信息以及客户端自身的信息。当我们打开一个网页时，浏览器要向网站服务器发送一个 HTTP 请求头，然后网站服务器根据 HTTP 请求头的内容生成当次请求的内容发送给浏览器，这就是 HTTP 请求报文中的“请求头”的作用。

HTTP 请求报文的“请求头部”是由一系列的行组成的，每行包括“头部字段名”和“值”这两个字段，它们之间用英文冒号“:”分隔，但也可以没有“请求头部”这部分。每一行的最后都有一个回车控制符和一个换行控制符（一起以 CRLF 表示），使下一个请求头在下一行显示。许多请求头都允许客户端指定多个可接受的“值”字段选项，有时甚至可以对这些选项的“值”字段选项进行排名，多个选项间以逗号分隔。典型的 HTTP 请求头如表 11-3 所示。

表 11-3 典型 HTTP 请求头

请求头	含义说明
Accept	指定客户能处理的 MIME（Multipurpose Internet Mail Extension，多用途互联网邮件扩展）页面类型。例如 Accept: image/gif，表明客户希望接受 GIF 图像格式的资源；Accept: text/html，表明客户端希望接受 html 文本
Accept-Charset	指定客户可以接受的字符集。例如 Accept-Charset:iso-8859-1,gb2312，表示客户可以接受的字符集在所列的两个标准中。如果在请求消息中没有设置这个请求头，则默认是任何字符集都可以接受
Accept-Encoding	指定客户能进行解码的数据编码方式。例如 Accept-Encoding:gzip, compress，表示客户可以接受 gzip 和 compress 这两种编码格式。如果请求消息中没有设置这个请求头，服务器假定客户可以接受各种内容编码

续表

请求头	含义说明
Accept-Language	指定客户能处理的语言类型。例如 Accept-Language:zh-cn，表示客户只接受中文简体语言。如果请求消息中没有设置这个请求头，服务器假定客户对各种语言都可以接受
Authorixation	指定客户信任的凭据列表。当浏览器访问一个页面时，如果收到服务器的响应代码为 401（未授权），可以发送一个包含 Authorization 请求报头域的请求，要求服务器对所列的用户账户进行验证
Cookie	将一个以前设置的 Cookie 送回给服务器
Connections	请求采用持续连接方式，例如 Connection:Keep-Alive
Date	指定消息发送时的日期和时间，例如 Date: Tue, 11 Jul 2012 18:23:51 GMT
From	向服务器请求一个邮箱地址，例如 From: winda@microsoft.com，这是假设笔者在访问微软网站时向微软的服务器申请的一个邮箱地址
Host	指定被请求服务器的域名和端口，如果是使用默认的 80 端口，则可以省略端口指定。例如 Host: www.51cto.com，表示请求的服务器域名为 www.51cto.com，且使用默认的 TCP 80 端口进行访问
Referer	包括一条 URI 信息，要求从指定的 URI 中访问当前请求的页面
User-Agent	允许客户端将它的操作系统、浏览器和其他属性告诉服务器。如我们上网登录论坛的时候，往往会看到一些欢迎信息，其中列出了你的操作系统的名称和版本，以及你所使用的浏览器的名称和版本，这就是 User-Agent 请求头的功劳了
Upgrade	客户希望切换到新的协议

3. 空行

在 HTTP 请求报文最后的一个请求头之后是一个空行，发送回车符和换行符（一起以 CRLF 表示），通知服务器以下不再有请求头。

4. 实体主体行

在请求报文中“实体主体行”部分通常是不用的，而且不能在 GET 方法中使用，仅在 POST 方法中用于向服务器提供一些用户凭据信息。

一个 HTTP 请求报文示例如下：

```
POST /servlet/default.jsp HTTP/1.1      !---这是一个 POST 方法请求行，指出了所附加的消息的 URL 和所用的 HTTP
                                        !---协议版本为 1.1
Accept-Language: zh-cn,zh     !---指定客户端可以接受中文简体和中文两种语言
    Accept-Charset: GB2312,utf-8      !---指定客户端可以接受的字符集为 GB2312 和 utf-8
    Accept: text/html,application/xhtml+xml,application/xml   !---指定客户端浏览器支持的 MIME 类型分别是
                                        !--- text/html、application/xhtml+xml、application/xml
    Accept-Encoding: gzip, deflate      !---指定客户端浏览器支持的压缩编码格式为 gzip 和 deflate
      User-Agent: Mozilla/5.0           !---指定客户端使用的浏览器版本为 Mozilla 5.0
      Host: www.51cto.com               !---指定客户端要访问的服务器域名为 www.51cto.com
    Connection: Keep-Alive              !---指定采用持续连接方式
（这里有一个空行）
user=winda&pwd=1234                     !---此行表示向服务器提交的用户账户为 winda，密码为 1234
```

11.2.9 HTTP 响应报文格式

在 Web 服务器收到客户端发来的 HTTP 请求报文后，通过服务器处理后会返回一个 HTTP 响应报文给请求的客户端，以告知客户端 Web 服务器对客户端请求所做出的处理，如是否允许此次请求的 HTTP 连接、出现了什么连接错误、错误代码和错误原因等。HTTP 响应报文也是由 4 个部分组成，分别是：响应（Response）行、响应报头（Response Header）行、空行和实体主体（Entity Body）行，如图 11-6 所示。

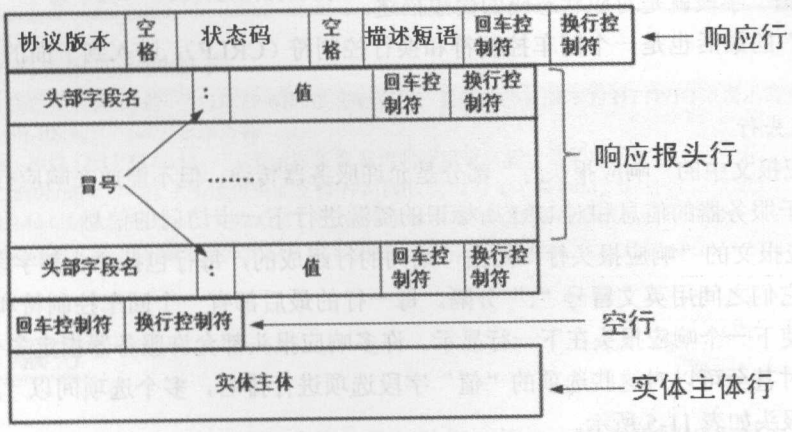


图 11-6 HTTP 响应报文格式

在 HTTP 响应报文中“空行”部分没有什么好介绍的，就是一个回车控制符和一个换行控制符（CRLF），其目的就是空一行显示下面的主体信息。“实体主体行”部分也基本不用。下面介绍“响应行”和“响应报头行”这两个部分。

1. 响应行

在 HTTP 响应报文的“响应行”中也主要是 3 个字段，分别是“协议版本”“状态码”和“描述短语”，它们之间用空格分隔。最后是各一个回车控制符和一个换行控制符（一起用 CRLF 表示）。其中“协议版本”字段没有什么好说的，就是显示 Web 服务器所使用的 HTTP 协议版本。

“状态码”字段比较重要，它是用一个三位数表示不同的状态，如请求是否被满足，如果没有满足的话原因是什么，共有 5 组取值，如表 11-4 所示。

表 11-4 HTTP 响应报文中的“状态码”类型

状态码类型	含义说明	示例
1xx	为指示类响应，表示请求已接收，继续处理	100=服务器同意处理客户请求
2xx	为成功类响应，表示请求已被成功接受	200=请求成功；204=无内容，也表示请求成功
3xx	为重定向类响应，表示要完成请求必须进行更进一步的操作	301=页面重定向了

续表

状态码类型	含义说明	示例
4xx	客户端错误类响应，表示客户端请求有语法错误或请求无法实现	400=客户请求有语法错误，不能被服务器所理解；401=请求未经授权；403=服务器收到请求，但拒绝提供服务；404=请求资源不存在
5xx	服务器端错误类响应，表示服务器未能实现所需要的请求	500=服务器发生了不可预期的错误；503=服务器当前不能处理客户端的请求，过一段时间再试

“描述短语”字段就是对应状态码的简短描述。

“响应行”的最后也是一个回车控制符和换行控制符（CRLF），进入到下面的“响应报头行”部分。

2. 响应报头行

HTTP 响应报文中的“响应报头行”部分是允许服务器传递，但不能放在响应行中的附加响应信息，以及关于服务器的信息和对 URI 所标识的资源进行下一步访问的信息。

HTTP 响应报文的“响应报头行”是由一系列的行组成的，每行包括“头部字段名”和“值”这两个字段，它们之间用英文冒号“:”分隔。每一行的最后都有一个回车控制符和一个换行控制符（CRLF），使下一个响应报头在下一行显示。许多响应报头都允许服务器指定多个返回的“值”字段选项，有时甚至可以对这些选项的“值”字段选项进行排名，多个选项间以“/”分隔。典型的 HTTP 响应报头如表 11-5 所示。

表 11-5　典型 HTTP 响应报头

响应报头	含义说明
Allow	显示服务器支持哪些请求方法
Server	显示服务器的软件信息。例如 Server: Apache-Coyote/1.1，表示 Web 服务器软件为 Apache-Coyote，版本为 1.1
Content-Encoding	显示请求文档所采用的编码方法。例如 Content-Encoding: gzip，表示文档采用的是 gzip 压缩方式
Content-Language	显示请求页面所使用的语言。例如 Content-Language: zh/zh-cn，表示页面支持中文和中文简体两种语言
Content-Length	显示请求页面的长度（以字节为单位）。例如 Content-Length: 1024，表示页面大小为 1024 个字节
Content-type	显示页面所支持的 MIME 类型。例如 Content-type: text/html，表示页面支持 text 和 html 两种消息类型
Date	指定消息发送时的日期和时间。例如 Date: Tue, 11 Jul 2012 18:23:51 GMT
Last-Modified	显示请求页面最后被编辑或修改的日期和时间。例如 Last-Modified: Tue, 11 Jul 2012 12:11:51 GMT
Location	指示客户将请求发送到所指定的位置，起到重定向的作用

续表

响应报头	含义说明
Accept-Range	显示服务器将接受指定字节范围内的请求。例如 Accept-Range: 1024，表示服务器只接受 1024 个字节以内的请求
Refresh	指示客户多少秒后再刷新或者重新访问本页面。例如 Refresh: 5，表示客户端要过 5 秒后再刷新或访问本页面
Set-Cookie	指示客户设置和页面关联的 Cookie
Upgrade	服务器希望切换到新的协议

一个 HTTP 响应报文示例如下：

```
HTTP/1.1 200 OK      !---这是一个 HTTP 响应报文响应行，显示的协议版本为 HTTP/1.1，表示请求成功
Server: Microsoft-IIS/6.0    !---表示服务器
Date: Sun, 3 Jan 2012 13:13:33 GMT    !---表示服务器发送此响应报文的日期和时间
Content-Type: text/html    !---指示所请求页面支持的 MIME 类型为 text 和 html
Last-Modified: Mon, 14 Apr 2012 13:23:42 GMT    !---指示所请求页面的最后修改日期和时间
Content-Length: 112    !---指示服务器接受前 112 个字节的请求内容
```

11.3 DNS 服务

DNS（Domain Name System，域名系统）是一种把计算机主机名称解析为对应的 IP 地址的服务。在 UNIX 和 Linux 操作系统中 DNS 服务通常被称为 BIND（Berkeley Internet Name Domain Service，伯克利因特网名称域服务）。但需要特别说明的是，从网络通信原理上来讲，DNS 并不是必需的，因为可以直接通过 IP 地址进行访问，而且事实上网络通信时最终所采用的寻址方式也是网络层的 IP 地址寻址。下面先来介绍一下 DNS 技术的一些引入背景。

11.3.1 DNS 技术的引入背景

说到为什么要引入 DNS 服务，我们先要回顾一下数据通信的基本原理。

我们在上一章介绍传输层服务时就提到了应用层的 Socket（套接字）服务。套接字是 IP 地址和端口号的组合，中间用冒号或逗号分隔，如（192,168.10,80）。每个 TCP 传输连接有两个端点，也就是有两个套接字，即“源 IP 地址和端口号”组合和“目的 IP 地址和端口号”组合，可以表示为｛socket1,socket2｝或者｛(IP1,Port1),(IP2,Port2)｝。其中的“IP 地址”是主机的网络层地址，用来在局域网或广域网中对各节点进行唯一标识（当然局域网和广域网可使用的 IP 地址范围是不一样的，具体参见第 9 章有关局域网 IP 地址和公网 IP 地址的相关内容），而“端口号”则是为每个应用进程所分配的传输层地址。因为同一时间两台主机间可以进行多个网络应用，每个进程需要用一个端口来进行唯一区分（因为此时的多个应用进程中的源 IP 地址和目的 IP 地址都是一样的）。

1. IP 地址标识的不足

我们在应用层要访问目的主机时，不仅要指出该目的主机的网络层 IP 地址（用于网络寻址），还要指出在传输连接中所使用的端口号以及在应用层所使用的应用协议。如我们在访问 Web 服务器时就需要在用户浏览器的地址栏中输入类似于 http://118.144.78.54:80 之类的地址，其中 http 就是我们访问 Web 服务器时所使用的应用层 HTTP 协议，118.144.78.54 是对应 Web 服务器的 IP 地址，80 就是对应 Web 服务器应用进程传输层端口号（当然，如果是常规服务，这个端口号是不用输入的）。

从以上介绍可以看出，使用 IP 地址来进行各种网络应用是完全没有问题的。但是那些位于互联网上，供全球用户公开访问的各种公用服务器来说，这种 IP 地址标识方式存在以下几个方面的不足：

（1）不便记忆。

尽管我们现在是把 IP 地址以比较简单的十进制表示，在浏览器的地址栏中也是以十进制格式 IP 地址表示的，其长度也不是很长（十进制 IPv4 地址最长也仅 12 位），但是它远没有普通的名称好记，就像要记一个人的身份证号要远比记一个人的名字困难一样，哪怕身份证号只有短短的几位。

当然以上是针对现在仍主流使用的 IPv4 地址来说的，而对于长度为现行 IPv4 地址 4 倍的 IPv6 地址来说这方面的不足就更为明显了。

（2）不便于地址变更。

Web 服务器，特别是互联网上的 Web 服务器的 IP 地址可能会因各种原因而变更。如果采用 IP 地址进行标识的话，每次 IP 地址的变更对这种开放型的互联网服务器来说可能带来致命的打击，因为那么多用户已知或未知的用户，我们不可能都一一通知到。但如果采用的是网站服务器名称来进行标识，IP 地址怎么变化都没有影响，只要我们的服务器主机名称不变。

（3）不安全。

现在互联网上非常不安全，如果直接把网站服务器的 IP 地址对外暴露，显然不是个安全的做法，就像我们的身份证号最好不要对外公布一样，因为这很可能被一些别有用心的人利用（窃取公网 IP 地址、盗链接等）。尽管对一些专业黑客来说，无论如何做好安全防范工作，要想获取服务器 IP 地址都是一件很容易的事，但防范总好过不防范，至少是在心理上。

2. Host 文件名称解析方案

为了克服以上 IP 地址标识方案的不足，于是有人想到用更方便记忆的服务器名称来替代服务器 IP 地址。在 1987 年引入 DNS 之前的 APPNET（互联网的前身）中是通过一个称为 host.txt 的文本文件把网络中各计算机的计算机名称与其对应的 IP 地址一一列出来实现解析的。在 Host 文件的 IP 地址和计算机名映射表项中，前面是 IP 地址，后面是所对应的计算机主机名（最初仅是像我们通常所说的 NetBIOS 格式计算机名），每个映射独占一行，格式如下（以下仅为示例，并非真实的映射）：

```
127.0.0.1    localhost      !---此为本地计算机环路映射
196.168.1.109   microsoftdesk
202.16.83.15    sunfinal
……
```

然后由各计算机调用这个 Host 文件以实现从计算机名到 IP 地址的解析，这个 Host 文件当时是由专门一台或多台服务器进行管理的（其实在 Windows、Linux 和 UNIX 系统主机上也有这个文件，仅用于本地计算机上的名称解析）。当然这是在没有使用 DNS 域名情况下的映射（主要用于局域网中），在使用了 DNS 域名时，后面的计算机名也可以是 DNS 域名格式。

随着互联网上计算机的指数级增加，人们发现以上这种通过 Host 文件进行名称解析的方案至少存在以下两个方面的不足：

（1）名称冲突在所难免。

因为没有一种可以限制名称冲突的机制，所以各公司在为自己的服务器起名时就难免与其他公司的服务器计算机名发生冲突。就像我们每个人的姓名一样，在全国范围来讲，与我们同名的人不知道有多少，因为没有规定不允许同名。

（2）少数名称解析服务器难以承受。

随着互联网上服务器数的飞速增加，仅靠授权机构中少数通过调用 Host 文件进行名称解析的名称服务器根本无法承受如此大的负荷。

3. DNS 名称解析方案

为了解决 Host 文件名称解析方案的以上不足，开发出了基于域（这里的“域”是指“域名”）的分层命名方案——DNS。它也可以将主机名映射成对应的 IP 地址。其基本的名称解析原理如下：当一个应用进程需要把主机名解析为 IP 地址时，该应用程序就调用一个解析器（Resolver），使它成为 DNS 客户，将该名字作为 DNS 请求报文参数，以 UDP 用户数据报方式发送给本地 DNS 服务器；然后本地 DNS 服务器查找该名字，并且将找到的 IP 地址放在响应报文中返回给解析器；解析器再将 IP 地址返回给调用解析器的应用进程，这样应用进程就可以根据所得到的目的 IP 地址进行通信了。当然，可能会存在本地 DNS 服务器不能解析所请求的名字的情况，这时本地 DNS 服务器就仅为一个 DNS 代理角色，然后它可以把这个解析工作交给在本地服务器上配置的其他相关联的 DNS 服务器，这就是 DNS 服务器的“转发器”功能。

有关 DNS 详细的命名方案将在下一节中介绍。

11.3.2 DNS 命名方案的设计思想

从设计思想上来看，开发 DNS 命名方案就是想通过某种方法解决上一节提到的 Host 名称解析方案中“容易发生名称冲突”和“服务器解析性能不足”这两方面的问题。这个方案就是采用基于域名的分层命名方案，下面具体分析这一命名方案是如何解决这两方面问题的。

1. 名称冲突的解决方案

目前，计算机名主要有两种格式：一种是像前面所说的早期 Host 文件中那样的非分级的单一字符串所代表的本地计算机名称，如 webserver；另一种是以 DNS 域名为基础的域网络中的 DNS 名称，如 webserver.lycb.com，后面的 lycb.com 就是一个域名。完整的 webserver.lycb.com DNS 名称所代表的就是在 lycb.com 域中的一台名为 webserver 的服务器。

【注意】其实 DNS 域名的最后面还有一个小圆点，代表的是根域名，只是平时我们在书写时

不写而已。另外，DNS 域名不仅应用于 Web 网站标识，像 FTP 站点和电子邮件系统都是使用 DNS 域名的，只不过具体使用时前面所指定的协议或格式是不一样的：Web 网站使用的是 http 协议（如 http://www.itct.com.cn），FTP 站点使用的是 ftp 协议（如 ftp://www.itct.com.cn），电子邮箱地址格式是“邮箱用户账户名@邮箱 DNS 域名”（如 winda@itct.com.cn）。

从以上介绍可以看出，DNS 名称是由两大部分组成的，最前面那部分所代表的是服务器的本地计算机名称，后面全部是对应服务器所在域的域名，而且这两部分中间是以小圆点（.）分隔、连接的，即“本地计算机名.DNS 域名”。这就是为什么说 DNS 是基于域命名方案的原因。

但 DNS 域名也不是单一结构的，而是包含至少两个层次的分级结构，因为单一结构的域名无法满足全球如此之多互联网用户的需求，就像尽管没有限制姓名长度，中国的汉字有上万个之多，但是面对一个 13 亿人的大国，还是很难做到完全不重名的，如果真有制度限制不能重名的话，则后面出生的人名字可能都非常难听或者都是他们自己不喜欢的了。DNS 域名也一样，如果只有一级结构，则后面用户所能申请的域名基本上都是很长，不能符合用户特征要求的了。

在分层结构的 DNS 域名中必须有一个是顶级域名，然后在这个顶级域名下面再申请或者注册二级、三级，甚至更多级别的域名，各级域名间同样是以小圆点（.）分隔、连接的。最右边这部分代表的是顶级域名，左边这部分代表的是子域名，而且是级别最低的域名写在最前面，级别最高的域名写在最后面。下级域名必须隶属于上级域名。如 itct.com.cn 这个 DNS 域名是一个三级域名，其中 itct 是三级域名（在本域名中级别最低），它隶属于其二级域名 com，而二级域名 com 又隶属于其一级域名（又称顶级域名）cn。在每个级别域名下面又可以有多个并列关系的下级域名，如顶级域名 cn 下面可以有 com、net、gov、edu 等这样的二级域名，而在 net 这样的二级域名下面同样可以有由各用户申请注册的三级域名。有关各级域名将在下一节中介绍。

设计这样的分层结构域名的好处很明显，就是上级域名下面可以有多个不同的下级域名，且整个 DNS 域名中只要有一级域名不一样，就可认为该域名与其他域名是不一样的。这样就可以比较好地确保互联网上域名的唯一性。如 mov.microsoft.com 和 mail.microsoft.com 这两个域名就是不同的域名，sina.com.cn 和 sina.com 也是两个不同的域名。

在这里说几句题外话，如果我们国家规定人的姓名也采用这种起名方式（如基于家族的起名方式）的话，则可以确保全国范围内都没有重名，根本不用身份证了。国外好像是采用这样类似的起名方案的，所以我们看到许多国外的人名似乎一样，但完整的名字却是不一样的。如美国的老、少布什总统，他们的完整姓名就分成几个部分，分别为 George Herbert Walker Bush（乔治·赫伯特·沃克·布什）和 George Walker Bush（乔治·沃克·布什）。看起来是不是像 DNS 域名一样呢？

2. 解析性能不足的解决方案

在早期的 Host 文件计算机名到 IP 地址映射解决方案中，网络中的名称解析全是以少数几个集中存储 Host 文件的服务器主机来担当。随着网络规模的增大，这种完全集中的管理方式显然不能满足解析性能的要求，在今天的互联网上更是不可能实现的。

有了 DNS 的分层命名方案后，就可以在 Host 文件集中管理的基础上进一步采取分布式的管理

方式。也就是可以针对每一级域名用少数的 DNS 服务器单独进行名称解析，这样一来，各 DNS 服务器的解析压力就不再难以承受了。如 www.microsoft.com 这个 DNS 名称中，顶级域名 com 是由对应的顶级域名提供商的 DNS 服务器负责解析的，二级域名 microsoft 是由对应的二级域名提供商的 DNS 服务器负责解析的。

目前整个互联网上可以注册的域名非常之多，而且每一类域名又可以由许多域名提供商提供，这样一来，即使是同一类域名，在全球范围内都可以由非常多分布在全球不同国家、地区的域名提供商的 DNS 服务器来共同担当名称解析服务，所以不再存在性能不足的问题。但需要特别说明的一点是，这些 DNS 服务器都不是孤立的，即使是完全不同的域名解析服务器，它们之间都有一定的从属关系，这与域名提供商的级别有关。下级域名提供商的 DNS 服务器必须与上级域名提供商的 DNS 服务器建立某种关联，否则可能同一域名会在不同地区被多人注册了，这是不允许的。

11.3.3 DNS 名称空间

从大的范围讲，整个互联网就是一个 DNS 名称空间，这个空间就是由当时可以注册的各级域名共同构成的。当然，这个空间会随着用户的需求和互联网域名管理机构（ICANN 或 NeuLevel）颁布的可注册域名数量和类型的扩展而扩展。整个 DNS 名称空间像一棵倒放的树（如图 11-7 所示），最顶端称为互联网的“根域”，以一个小圆点(.)表示，下面是顶级域，再下面是二级域，三级域，……，依此类推（一般最多是三级，注意域名是不区分大小写的，但通常是小写）。其中顶级域名和二级域名必须是由对应的互联网域名管理机构颁发的。

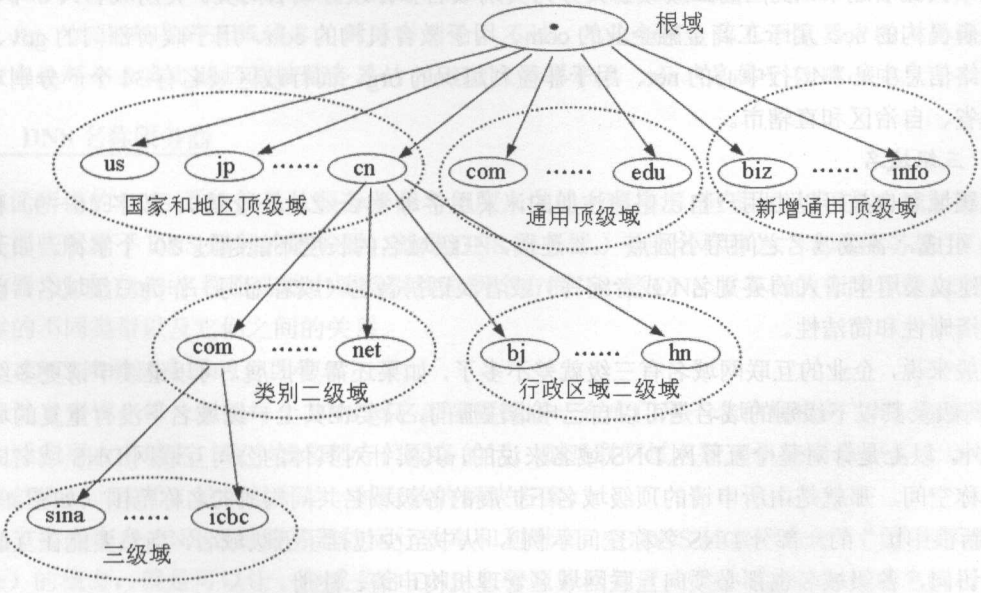


图 11-7 互联网上的 DNS 名称空间

目前，互联网上注册的DNS域名主要是3个级别：顶级域名、二级域名和三级域名，其中只有二级及其以下级别的域名才可由用户自己申请，用户是不能直接申请顶级域名的。

1. 顶级域名

顶级域名是根域名（.）下面的第一级域名，但它不能单独为用户分配。目前顶级域名有三大类：一类是国家和地区类的，一类是通用类的，还有一种是近期新增的通用类的。国家和地区类顶级域名是为每个国家和国际独立的地区分配的域名，在ISO3166中规定，cn代表中国，us代表美国，jp代表日本，tw代表我国台湾地区。

通用类顶级域名在RFC1591中规定，如商业类的com、教育类的edu、政府类的gov、国际组织类的int、网络供应商类的net、非盈利组织类的org等。近期新增的通用类顶级域名有biz（商业组织）、film（公司企业）、store（商店）、web（突出WWW活动的组织）、arts（突出文化、娱乐活动的组织）、rec（突出消遣、娱乐活动的组织）、info（提供信息服务的组织）、.nom（个人）。其中biz顶级域名是为了接替com域名的，因为com域名空间已消耗得差不多了，但它不再是由ICANN管理，而是由NeuLevel公司单独管理。

2. 二级域名

二级域名是指顶级域名之下的域名。它又分为两大类：在国际顶级域名下，它是指域名注册人的网上名称，例如ibm、yahoo、microsoft等，这是可供用户申请的；在国家顶级域名下，它是表示注册企业类别的符号，例如com、edu、gov、net等，这些是不能直接供用户申请注册的。我国在国际互联网络信息中心（Inter NIC）正式注册并运行的顶级域名是cn，这也是我国的一级域名。目前我国有中文域名，其中一级域名为“中国”。

在顶级域名之下，我国的二级域名又分为类别域名和行政区域名两类。类别域名共6个，包括用于科研机构的ac、用于工商金融企业的com、用于教育机构的edu、用于政府部门的gov、用于互联网络信息中心和运行中心的net、用于非盈利组织的org。而行政区域名有34个，分别对应于我国各省、自治区和直辖市。

3. 三级域名

三级域名也是可以由用户自己申请注册的，采用字母（A～Z、a～z等）、数字（0～9）和连接符（-）组成，各级域名之间用小圆点（.）连接，三级域名的长度不能超过20个字符。如无特殊原因，建议采用申请人的英文名（或者缩写）或者汉语拼音名（或者缩写）作为三级域名，以保持域名的清晰性和简洁性。

一般来说，企业的互联网域名有三级就差不多了，如果还需要扩展，可以继续申请更多级别的域名。三级及其以下级别的域名是可以自己申请注册的，只要在其上一级域名下没有重复的域名即可。另外，以上是针对整个互联网DNS域名来说的，其实针对具体的公司互联网DNS域名同样有一个名称空间，那就是由所申请的顶级域名下扩展的各级域名共同构成的名称范围（如图11-8所示是“新浪中国”的一部分DNS名称空间示例），从中至少包括了五级域名，当然要能在互联网中供用户访问，各级域名也都必须向互联网域名管理机构申请、注册。

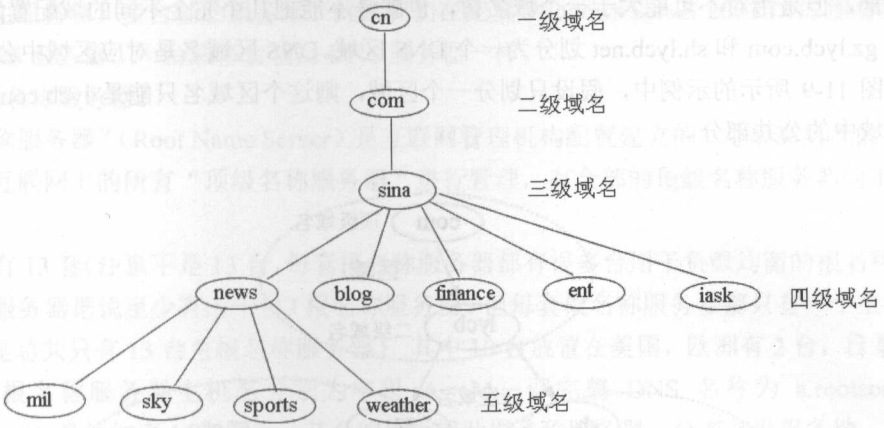


图 11-8 新浪中国的部分域名空间



所申请的 DNS 域名是不包括 NetBIOS 计算机名部分的，如 microsoft.com 是微软所申请的一个二级域名。在这个域名下可以有许多不同用途的主机，如 Web 服务器、FTP 服务器、E-mail 服务器等，而且它们都可以有多个同类服务器。这时像 www. microsoft.com、ftp. microsoft.com、mail. microsoft.com 之类的 DNS 名称中的 www、ftp、mail 就是服务器的 NetBIOS 计算机名，不包括在 DNS 域名之内。所以以上 DNS 名称都属于二级域名，而不是三级域名，当然如果你确实申请了像 www、ftp、mail 之类的三级域名，则另当别论。不过，无论如何，你在浏览器中输入的服务器地址中最左边那部分肯定就只是该服务器的 NetBIOS 计算机名，而不是属于 DNS 域名。

11.3.4 DNS 名称服务器

前面所说的 DNS 系统仅是从服务器命名角度来讲的一项技术，但真正担当 DNS 域名解析任务的还是那些配置了 DNS 服务的服务器，就是 DNS 服务器，专业名称称为“名称服务器”（或“域名服务器”）。在 DNS 名称服务器中要特别注意两个方面：一是 DNS 服务器的分区管理；二是 DNS 服务器的不同类型以及它们之间的关系。

1. DNS 名称服务器的分区管理

理论上来说，每一级的每个 DNS 域名都需要配备一台专门用于名称解析的 DNS 名称服务器，但是全球有那么多域名，每个域名都专门配备的话，则互联网域名管理机构所需配备的 DNS 名称服务器就实在太多了，这明显不是一种有效的解决方案。

为了提高每台 DNS 名称服务器的运行效率和利用率，在 DNS 技术中为此专门提出了“区域”（zone）的概念，就是可以让一台或多台 DNS 名称服务器负责一个区域的计算机域名解析，而这个“区域”可以包括一个 DNS 域名树（或者说“DNS 域名空间”）的一部分（通常是包括多级域

名）或全部，但是绝对不可能大于一个域名树，也就是不能把几个完全不同的域配置为一个区域，如不能把 gz.lycb.com 和 sh.lycb.net 划分为一个 DNS 区域。DNS 区域名是对应区域中公共的域名树部分，如图 11-9 所示的示例中，假设只划分一个区域，则这个区域名只能是 lycb.com，因为这才是整个区域中的公共部分。

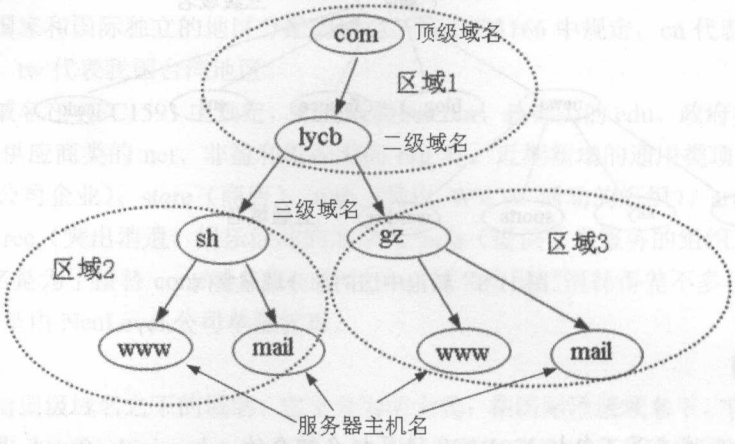


图 11-9　DNS 名称服务器分区示例

在如图 11-9 所示的 DNS 域名树示例中，如果为每一级的每个域名分别配置一个 DNS 名称服务器的话，则这样一个简单的域名树都需要 8 台 DNS 名称服务器，这显然是不合算的，也是没必要的。事实上，如果这个域网络中的应用服务器不是很多的话，整个 DNS 域就只划分一个区域（lycb.com），仅用一台或多台 DNS 服务器负责名称解析工作；如果各级域名中的应用服务器数比较多，则可以将整个 DNS 域名树分成如图中所示的 3 个区域，区域名分别为 lycb.com、sh.lycb.com 和 gz.lycb.com，然后为这 3 个区域各配置一台或多台 DNS 名称服务器即可。这就是 DNS 名称服务器的分区理念。

至于具体要把哪几级域名划分在同一个区域，这个没有固定规则，完全由管理员来决定。一般是取决于各级域名中应用服务器的数量以及这些服务器的物理位置。通常是尽可能使各区域中应用服务器的数量均衡，以便使各区域中的 DNS 名称服务器的解析负担尽可能均衡。同时还尽可能把物理位置相近的应用服务器划分到同一个区域中。当然，越上一级别区域的 DNS 名称服务器的负担越重，因为它同时还会为下级区域中的 DNS 名称服务器提供解析支持，在下级区域中的 DNS 名称服务器解析不了或者满负荷时提供帮助。

2. DNS 名称服务器种类

为了有效管理整个互联网的 DNS 域名解析工作，DNS 系统开发者也设计了一个与分层的 DNS 域名结构类似的层次化 DNS 名称服务器结构。把所有 DNS 名称服务器自高到低分成 4 个级别：根名称服务器、顶级名称服务器、权威名称服务器和本地名称服务器，它们都是由互联网域名管理机构或下级的 ISP 负责配置建立的，但它们都不是由一台服务器担当，即使是一个具体的域名服务器

也都有一台主服务器，多台辅助服务器。就像我们在配置 Windows Server 2003 DNS 服务器时要配置一台主 DNS 服务器，一台或多台辅助 DNS 服务器一样。

（1）根名称服务器。

“根名称服务器”（Root Name Server）是互联网管理机构配置建立的，是最高层次的名称服务器，负责对互联网上的所有“顶级名称服务器”进行管理，有全部的顶级名称服务器的 IP 地址和域名映射。

全球共有 13 套(注意不是 13 台,每套根名称服务器都有很多台用于负载均衡的根名称服务器，总的根名称服务器据说至少有上千台）根名称服务器。但每套根名称服务器都只有一个主根名称服务器（也就是总共只有 13 台主根名称服务器），其中 10 台放置在美国，欧洲有 2 台，日本有 1 台。这 13 台主根名称服务器主机名分别为字母 A～M，即完整 DNS 名称为 a.rootserver.net～m.rootserver.net，具体如表 11-6 所示。其余的均为辅助根名称服务器，分布在世界各地，如中国地区有 3 台，分别位于北京、香港和台湾，目的是为了让世界各地都能就近获取根名称服务器的解析。任何域名解析都要经过这 13 套根名称服务器获得顶级名称服务器索引。

表 11-6　13 台主根名称服务器的管理机构和 IP 地址

名称服务器主机名	管理单位及设置地点	IP 地址
A	INTERNIC.NET（美国，弗吉尼亚州）	198.41.0.4
B	美国信息科学研究所（美国，加利弗尼亚州）	128.9.0.107
C	PSINet 公司（美国，弗吉尼亚州）	192.33.4.12
D	马里兰大学（美国，马里兰州）	128.8.10.90
E	美国航空航天管理局（美国，加利弗尼亚州）	192.203.230.10
F	因特网软件联盟（美国，加利弗尼亚州）	192.5.5.241
G	美国国防部网络信息中心（美国，弗吉尼亚州）	192.112.36.4
H	美国陆军研究所（美国，马里兰州）	128.63.2.53
I	Autonomica 公司（瑞典，斯德哥尔摩）	192.36.148.17
J	VeriSign 公司（美国，弗吉尼亚州）	192.58.128.30
K	RIPE NCC（英国，伦敦）	193.0.14.129
L	IANA（美国，弗吉尼亚州）	198.32.64.12
M	WIDE Project（日本，东京）	202.12.27.33

这些根名称服务器其实在我们配置 Windows Server 2003 DNS 服务器的“根提示”时就可以见到，如图 11-10 所示。从中也可以见到如表 11-6 中所示的这些根名称服务器的 IP 地址。当然，如果你配置的是局域网的域名，这些根名称服务器是用不上的，因为这些根名称服务器仅用于互联网上的域名解析。有许多读者朋友一直不理解在 Windows Server 2003 DNS 服务器配置“根提示”的用途，现在相信已经明白了吧。

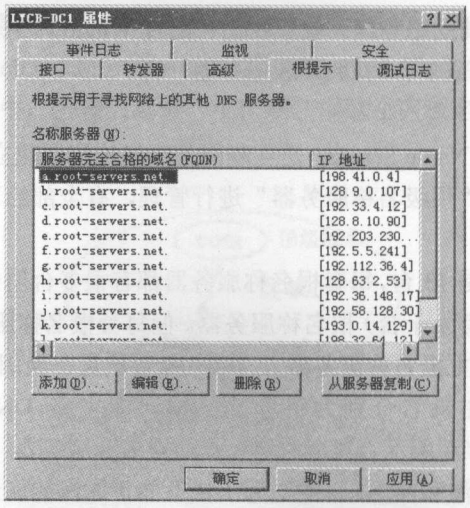


图 11-10　Windows Server 2003 DNS 服务器中的根名称服务器配置对话框

【注意】根名称服务器并不直接用于名称解析，因为这些根名称服务器上也没有保存全部的互联网域名记录（仅有负责管理顶级域名的顶级域名服务器的相关记录）。根名称服务器的作用仅是当用户本地名称服务器解析不了某个顶级域名时告诉本地名称服务器去找哪个顶级名称服务器。

（2）顶级名称服务器。

“顶级名称服务器”（Top Level Name Server）顾名思义就是各顶级域名自己的名称服务器，负责它们各自所管理的顶级域名解析，并通常还具有二级名称服务器的地址信息。每一个顶级域名，不管是 gTLD（通用顶级域）还是 ccTLD（国别顶级域），它们都有自己的域名服务器。如顶级域名 com 和 net 有 13 台名称服务器，主机名也是从 A 到 M，域名为 gtld-servers.net；顶级域名 biz 有 6 台名称服务器，主机名是从 A 到 H，域名为 gtld.biz。

（3）权威名称服务器。

“权威名称服务器”（Authoritative Name Server）是针对前面所说的 DNS 区域提供名称解析服务而专门配置、建立的名称服务器，可为用户提供最权威的 DNS 域名解析。每一个域名在互联网上都可以找到一台权威名称服务器，ISP 也可为用户的每一个 DNS 域名区域配置一台权威名称服务器。

（4）本地名称服务器。

这里所说的“本地名称服务器”不是指用户局域网中的名称服务器，而是用户端操作系统所配置的，由本地 ISP 提供的名称服务器（也就是本地 DNS 服务器）。它是离用户最近的互联网名称服务器。用户发出的 DNS 域名解析请求首先到达的就是本地名称服务器。

如果本地名称服务器解析不了用户所请求的域名时，这个本地名称服务器就会直接向所配置的根名称服务器发出解析请求，由根名称服务器告知它该向哪个顶级名称服务器查询；如果顶级名称服务器还不能解析时，顶级名称服务器会向本地名称服务器告知要向哪个权威名称服务器请求，然

后本地名称服务器继续向对应权威名称服务器解析；权威名称服务器还不能解析的话，则这次 DNS 域名解析请求失败。有关名称解析原理将在本章后面介绍。

11.3.5 DNS 报文格式

DNS 报文与 HTTP 报文一样，也分请求报文和应答报文两类，但 DNS 的请求报文和应答报文的总体格式是一样的，只是一些参数的取值不一样而已。DNS 报文也分两部分：报头部分和数据部分，如图 11-11 所示。

【说明】“应答消息”“授权应答”和“附加信息”这三部分仅应答报文有，“查询消息”仅请求报文有。

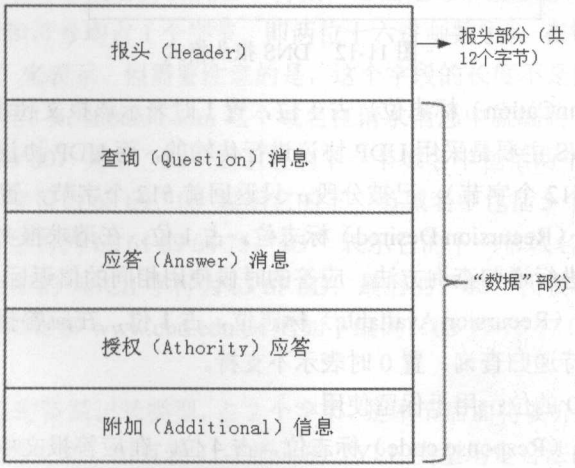


图 11-11 DNS 报文基本格式

1. DNS 报头

DNS 报头格式如图 11-12 所示，各字段的解释如下：

- ID：这是由 DNS 查询程序指定的 16 位请求标识符，每次查询均会分配一个标识符，相当于查询报文序号一样。该标识符也被随后的应答报文所用，也就是应答报文中的标识符与其对应的请求报文中的标识符是一样的，这样即可区分应答消息所对应的请求消息。
- QR：报文类型标志位，占 1 位，请求报文置 0，应答报文置 1。
- OPcode：操作码标志位，用于设置查询的种类，占 4 位，应答的时候会带相同值。置 0 时表示为标准查询（QUERY）；置 1 时表示为反向查询（IQUERY）；置 2 时表示为服务器状态查询（STATUS）；其他值保留，暂时未使用。
- AA：授权应答（Authoritative Answer）标志位，占 1 位，仅在应答报文中有意义。置 1 时表示在应答报文中所给出的名称服务器是所查询域名的权威名称服务器。

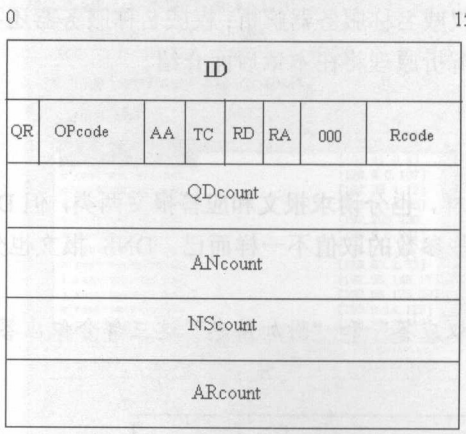


图 11-12　DNS 报头格式

- TC：截断（TrunCation）标志位，占 1 位，置 1 时表示该报文超出最大报文长度（512 个字节，因为 DNS 主要是采用 UDP 协议进行传输的，而 UDP 协议在互联网上通常允许传输的 MTU 为 512 个字节），已被分段，只返回前 512 个字节，置 0 时表示未被分段。
- RD：期望递归（Recursion Desired）标志位，占 1 位，在请求报文中设置，置 1 时表示建议域名服务器进行递归查询方法，应答的时候使用相同的值返回。
- RA：支持递归（Recursion Available）标志位，占 1 位，在应答报文中设置，置 1 时表示名称服务器支持递归查询，置 0 时表示不支持。
- 后面是 3 个置 0 的位：用于保留使用。
- Rcode：应答码（Response code）标志位，占 4 位，在应答报文中设置，置 0 时表示无错误；置 1 时表示报文格式有错误，名称服务器不接受请求的报文；置 2 时表示是因为服务器的原因而导致无法处理解析请求；置 3 时表示解析的域名不存在；置 4 时表示名称服务器不支持所请求的查询类型；置 5 时表示名称服务器由于设置的策略而拒绝给出应答。比如，服务器不希望对某些请求者给出应答或者服务器不希望进行某些操作（比如区域传输（zone transfer））；其他值保留，暂时未使用。
- QDcount：16 位整数，标识“查询消息”字段中的问题条数，仅请求报文中有。
- ANcount：16 位整数，标识“应答消息”字段中的资源记录数，仅应答报文中有。
- NScount：16 位整数，标识“授权应答”字段中的名称服务器资源记录数，仅应答报文中有。
- ARcount：16 位整数，标识“附加信息”字段中的资源记录数，仅应答报文中有。

2. 请求消息格式

DNS 请求报文中的具体请求消息在如图 11-11 所示的报文格式“查询消息”部分显示，包括对应 DNS 域名查询所请求的问题。每条查询消息的格式如图 11-13 所示。

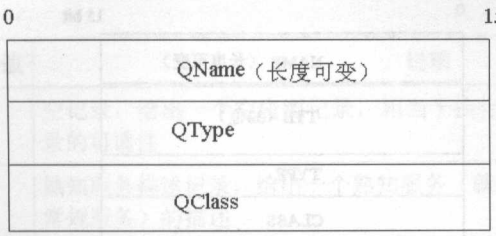


图 11-13 查询消息格式

- QName：所请求解析的域名，不过它显示的是一串 ASCII 编码，长度可变（注意不是固定的 16 位）。这里比较难理解。在每个域名的开头和中间的小圆点（.）部分均用一个十六进制数表示下一节域名的 ASCII 字符数，后面是每节域名各个符号所对应的 ASCII 字符（每个字母和符号均占 1 个字节，即两位十六进制数字），在每个域名的最后均以一个字节的 0（00）来表示。但需要注意的是，这个字段的长度不是固定的，也可以为奇数个字节，不用填充。如 hi.baidu.com 这个域名在请求消息中就编码为 02 68 69 05 62 61 69 64 75 03 63 6F 6D 00，其中“02”表示它的下一节域名中包括两个 ASCII 字符，即“hi”，对应的 ASCII 字符为 68 69;“05”表示它的下一节域名中包括 5 个 ASCII 字符，即“baidu”，对应的 ASCII 字符为 62 61 69 64 75;“03” 表示它的下一节域名中包括 3 个 ASCII 字符，即“com”，对应的 ASCII 字符为 63 6F 6D；最后的“00”代表该域名的结束。用同样的方法我们就可以理解 www.chd.edu.cn 的如下编码：03 77 77 77 03 63 68 64 03 65 64 75 02 63 6E 00。
- QType：所查询的资源记录类型，占 2 个字节，是本节后面将要介绍的资源记录中的 TYPE（类型）字段的扩展。所有表 11-7 所示的 TYPE 类型均是合法的 Qtype 取值（有些通用的 QType 值可以与多条资源记录相匹配），另外还可以有以下取值：
  - AXFR：Authoritative Transfer（权威转换器），值为 252，代表完整区域的域名解析请求。
  - MAILB：Mail Box（邮箱），值为 253，代表与邮箱相关的资源记录，如 MB（邮箱）、MG（邮件组）或者 MR（邮箱名更改）记录。
  - MAILA：Mail Agent（邮件代理），值为 254，代表邮件代理资源记录，如目前已基本不用的 MX（邮件交换）记录。
  - *：值为 255，代表所有资源记录的请求。
- QClass：查询类别，占 2 个字节，是本节后面将要介绍的资源记录中的 CLASS（类别）字段的扩展。所有表 11-8 所示的 CLASS 字段值均是合法的 QClass 字段取值。另外新增了一个“*”查询类别，取值为 255，代表所有分类。

3. 资源记录格式

在 DNS 应答报文中，会在如图 11-11 所示的“应答消息”“授权应答”和“附加信息”部分显示所应答的资源记录信息。这些资源记录信息格式是一样的，如图 11-14 所示。

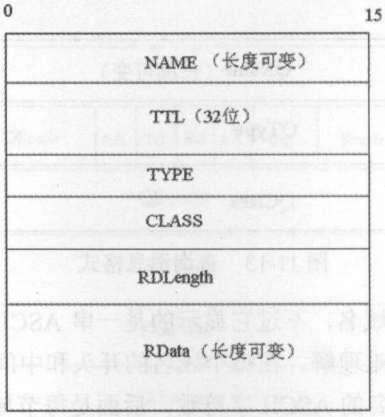


图 11-14　资源记录格式

各字段说明如下：

- NAME：资源记录对应的域名，与 DNS 请求报文 QName 字段所请求域名的一致，该字段长度可变。
- TTL：表示该资源记录可以缓存的时间（以秒为单位），一般用于当地址解析程序取出资源记录后决定保存及使用缓存数据的时间，占 4 个字节，置 0 时代表只能被传输不能被缓存。
- TYPE：资源记录类型，占 2 个字节。可用的资源记录类型及取值说明如表 11-7 所示。这里仅给出对应资源记录的类型，并不是具体的资源记录内容，具体的资源记录内容在后面将要介绍的 RData 字段中。

表 11-7　TYPE 值类型及取值说明

资源记录类型	对应的字段值	说明
A	1	主机记录，给出一个主机 IP 地址，配置了多个 IP 地址的主机有对应数量的 A 记录
NS	2	名称服务器记录，给出一个权威名称服务器 IP 地址
MD	3	邮件目标记录，给出一个目的邮件地址，已过时，现用 MX 记录替代
MF	4	邮件转发器记录，给出一个邮件转发器，已过时，现用 MX 记录替代
CNAME	5	规范名记录，给出一个别名的规范名称
SOA	6	起始授权机构记录，给出一个区域的起始授权机构名称服务器 IP 地址
MB	7	邮箱记录，给出一个邮箱域名，仅用于实验用途
MG	8	邮件组记录，给出一个邮件组成员，仅用于实验用途
MR	9	邮件更名记录，给出一个邮件重命名后的域名，仅用于实验用途

续表

资源记录类型	对应的字段值	说明
NULL	10	空记录，给出一个空资源记录，相当于换行，主要是为了增加整个资源记录的可读性
WKS	11	熟知服务描述记录，给出一个熟知服务（就是像 HTTP、FTP、SMTP 这类常规服务）的描述
PTR	12	指针记录，给出一个 IP 地址的别名
HINFO	13	主机信息记录，给出主机操作系统和 CPU 信息
MINFO	14	邮箱信息记录，给出一个邮箱或邮件列表信息
MX	15	邮件交换记录，给出邮件交换的优先级以及希望接受该域电子邮件的主机
TXT	16	文本记录，一本文本字符串

- CLASS：资源记录数据的类别，占 2 个字节，可用的数据类别及取值说明如表 11-8 所示，通常都是 IN（即 Internet）类别。这里仅给出对应资源记录数据（也就是 RData 字段中的内容）的类别，并不是具体的资源记录内容，具体的资源记录内容在后面将要介绍的 Rdata 字段中。

表 11-8 CLASS 值类别及取值说明

资源记录数据类别	值	说明
IN	1	给出请求解析域名所对应的 IP 地址，是默认的资源记录数据类别
CS	2	CSNET（Computer Science Network，计算机科学网络）类别，目前基本不用
CH	3	Chaos 类别，由以前的 Symbolics Lisp 机器使用，已经被废弃
HS	4	Hesiod 类别，用来查询用户主目录，现在也不用了

- RDLength：RData（资源数据）字段的长度（以字节为单位），占 2 个字节。
- RData：具体的相关资源记录，长度可变。其取值与 TYPE、CLASS 字段值有关。比如，TYPE 值为 A，CLASS 值为 IN，那么 RData 就是一个 4 个字节的 IP 地址。

11.3.6 DNS 数据传输方式

DNS 服务同样是 C/S 工作模式，分为 DNS 服务器和 DNS 客户端。DNS 客户端以 DNS 请求报文向 DNS 服务器发出域名解析请求，DNS 服务器以 DNS 应答报文对客户端的 DNS 请求做出应答。

DNS 既可以报文方式通过 TCP 53 号端口进行报文分组传输，也可以数据报方式通过 UDP 53 号端口进行数据报传输。至于什么时候使用 TCP 协议，什么时候使用 UDP 协议进行数据传输，这要从以下几个方面来考虑：

- 当 DNS 数据（包括报头和数据部分）大于 512 个字节时，只能采用 TCP 协议进行数据

传输，因为UDP中传输的数据最长不能超过512个字节。至于是否会分段，由具体网络中数据链路层的MTU值决定。如果被分段，则在DNS报头TC标志位置1。

- 在区域传输过程中，也就是在从主DNS服务器向辅助DNS服务器传输数据时必须使用TCP协议，因为这样更可靠。
- 当DNS数据小于512个字节时（这类情况比较少）具体使用TCP还是UDP是由DNS解析器决定的，但请求和应答均使用相同的协议，不会出现请求时使用的是TCP协议，而在应答时使用的是UDP协议的情况。一般递归解析时使用UDP协议。

DNS域名解析其实大多数情况是查找对应域名权威名称服务器的过程，除非在本地名称服务器缓存中已有对应域名的记录。但事实上名称服务器上的缓存容量和可以缓存的时间都有限，所以绝大多数情况下是需要权威名称服务器来帮助解析的。

DNS有两种名称解析方式：一种称为“递归解析”（Recursive Solution，RS），另一种称为“迭代解析”（Iterative Solution，IS，也称“反复解析”）。它们有不同的查询和应答流程：递归查询是一种代理查询方式，如果本地名称服务器不能解析，则后面的查询请求全由本地名称服务器代理DNS客户端进行查询，最终由本地名称服务器返回最后的解析结果；迭代解析中的查询请求则全由DNS客户端自己根据每次查询从上级DNS服务器上得到的信息依次查询下级DNS服务器。这就好比你要找一家你从未去过的公司，如果是通过一个服务公司为你去查找，然后把结果告诉你，这就相当于递归查询，而如果你是自己一步步去找人问，然后根据每次问话得到的信息再去找其他人问，这就是迭代查询。下面两节将分别介绍这两种名称解析方式。

11.3.7 DNS递归解析原理

“递归解析”（也叫“递归查询”，其实意思是一样的）是最常见，也是默认的解析方式。在这种解析方式中，如果客户端配置的本地名称服务器不能解析，则后面的查询全由本地名称服务器代替DNS客户端进行查询，直到本地名称服务器从权威名称服务器得到了正确的解析结果，然后由本地名称服务器告诉DNS客户端查询的结果。

1. DNS递归解析基本流程

在这个查询过程中，一直是以本地名称服务器为中心的，DNS客户端只是发出原始的域名查询请求报文，然后就一直处于等待状态，直到本地名称服务器发来了最终的查询结果。此时的本地名称服务器就相当于中介代理的作用。如果考虑了本地名称服务器的缓存技术（也就是在DNS服务器上对一定数量的以前查询记录保存一定时间，这样后面查询同样的域名信息时就可以直接从缓存中调出来，以提高查询效率），则递归解析的基本流程如下：

（1）客户端向本机配置的本地名称服务器（在此仅以首选DNS服务器为例进行介绍，所配置其他备用DNS服务器的解析流程完全一样）发出DNS域名查询请求。

（2）本地名称服务器收到请求后，先查询本地的缓存，如果有该域名的记录项，则本地名称服务器就直接把查询的结果返回给客户端；如果本地缓存中没有该域名的记录，则本地名称服务器再以DNS客户端的角色发送与前面一样的DNS域名查询请求给根名称服务器。

（3）根名称服务器收到 DNS 请求后，把所查询到的所请求的 DNS 域名中顶级域名所对应的顶级名称服务器地址返回给本地名称服务器。

（4）本地名称服务器根据根名称服务器所返回的顶级名称服务器地址向对应的顶级名称服务器发送与前面一样的 DNS 域名查询请求。

（5）对应的顶级名称服务器在收到 DNS 查询请求后，也是先查询自己的缓存，如果有所请求的 DNS 域名的记录项，则把对应的记录项返回给本地名称服务器，然后再由本地名称服务器返回给 DNS 客户端，否则向本地名称服务器返回所请求的 DNS 域名中的二级域名所对应的二级名称服务器地址。

然后本地名称服务器继续按照前面介绍的方法一次次地向三级、四级名称服务器查询，直到最终的对应域名所在区域的权威名称服务器返回最终的记录给本地名称服务器。然后再由本地名称服务器返回给 DNS 客户，同时本地名称服务器会缓存本次查询得到的记录项。

2. DNS 递归解析示例

为了方便大家理解，下面举一个例子进行介绍。本示例中假设客户端想要访问自己并不识别的 example.microsoft.com 站点，并假设此客户端配置的本地名称服务器为 dns.company.com（通常是以 IP 地址方式配置的），本地名称服务器上配置的根名称服务器是 a.rootserver.net。整个递归名称解析的过程如图 11-15 所示。

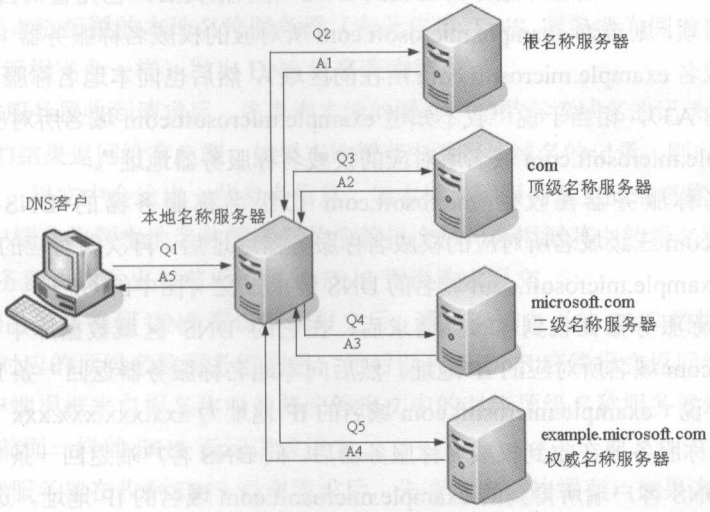


图 11-15 DNS 递归名称解析示例

其中的 Q1～Q5 表示发送 DNS 查询请求，A1～A5 是 DNS 查询应答，具体描述如下：

（1）DNS 客户端向所配置的本地名称服务器 dns.company.com 发出解析 example.microsoft.com 域名的 DNS 请求报文（图中的 Q1），相当于对本地名称服务器说“请给我 example.microsoft.com 所对应的 IP 地址”。

（2）本地名称服务器收到请求后，先查询本地缓存。假设没有查到该域名对应的记录，则本地名称服务器向所配置的根名称服务器 a.rootserver.net 发出请求解析 example.microsoft.com 域名的 DNS 请求报文（图中的 Q2）。

（3）根名称服务器收到查询请求后，通过查询得到.com 顶级域名所对应的顶级名称服务器，然后向本地名称服务器返回一条应答报文（图中的 A1），相当于说“我不知道 example.microsoft.com 域名所对应的 IP 地址，但我现在告诉你.com 域名所对应的顶级名称服务器地址”。

（4）本地名称服务器在收到根名称服务器的 DNS 应答报文，得到.com 顶级域名所对应的顶级名称服务器地址后，再次向对应的顶级名称服务器发送一条请求解析 example.microsoft.com 域名的 DNS 请求报文（图中的 Q3）。

（5）.com 顶级名称服务器在收到 DNS 请求报文后，先查询自己的缓存，假设也没有该域名的记录项，则查询 microsoft.com 所对应的二级名称服务器，然后也向本地名称服务器返回一条 DNS 应答报文（图中的 A2），相当于说“我不知道 example.microsoft.com 域名所对应的 IP 地址，但我现在告诉你 microsoft.com 域名所对应的二级名称服务器地址”。

（6）本地名称服务器在收到.com 顶级名称服务器的 DNS 应答报文，得到 microsoft.com 二级域名所对应的二级名称服务器地址后，再次向对应的二级名称服务器发送一条请求解析 example.microsoft.com 域名的 DNS 请求报文（图中的 Q4）。

（7）microsoft.com 二级名称服务器在收到 DNS 请求报文后，也先查询自己的缓存，假设也没有该域名的记录项，则查询 example.microsoft.com 所对应的权威名称服务器（因为这个名称服务器已包括了完整域名 example.microsoft.com 所在的区域），然后也向本地名称服务器返回一条 DNS 应答报文（图中的 A3），相当于说“我不知道 example.microsoft.com 域名所对应的 IP 地址，但我现在告诉你 example.microsoft.com 域名所对应的权威名称服务器地址”。

（8）本地名称服务器在收到 microsoft.com 二级名称服务器的 DNS 应答报文，得到 example.microsoft.com 三级域名所对应的权威名称服务器地址后，再次向对应的权威名称服务器发送一条请求解析 example.microsoft.com 域名的 DNS 请求报文（图中的 Q5）。

（9）权威名称服务器在收到 DNS 请求后，在它的 DNS 区域数据库中查找，最终得出了 example.microsoft.com 域名所对应的 IP 地址。然后向本地名称服务器返回一条 DNS 应答报文（图中的 A4），相当于说“example.microsoft.com 域名的 IP 地址为 xxx.xxx.xxx.xxx”。

（10）本地名称服务器在收到权威名称服务器后，向 DNS 客户端返回一条 DNS 应答报文（图中的 A5），告诉 DNS 客户端所得到的 example.microsoft.com 域名的 IP 地址。这样 DNS 客户端就可以正常访问这个网站了。

如果在步骤（9）中对应域名的权威名称服务器都说找不到对应的域名记录，则会向本地名称服务器返回一条查询失败的 DNS 应答报文，这条报文最终也会由本地名称服务器返回给 DNS 客户端。当然，如果这个权威名称服务器上配置了指向其他名称服务器的转发器，则权威名称服务器还会在转发器指向的名称服务器上进一步查询。另外，如果 DNS 客户端上配置了多个 DNS 服务器，则还会继续向其他 DNS 服务器查询。

11.3.8 DNS 迭代解析原理

在上一节介绍的 DNS 递归解析中，当所配置的本地名称服务器解析不了时，后面的查询工作是由本地名称服务器替代 DNS 客户端进行的（以“本地名称服务器”为中心），只需要本地名称服务器向 DNS 客户端返回最终的查询结果即可。而本节所介绍的 DNS 迭代解析（或者叫“迭代查询”）的所有查询工作全部是 DNS 客户端自己进行的（以“DNS 客户端”自己为中心）。在以下条件之一满足时就会采用迭代解析方式：

- 查询名称服务器时客户端的请求报文中没有申请使用递归查询，也就是说在 DNS 请求报头的 RD 字段没有置 1。相当于说，你都没有主动要求我为你进行递归查询，我当然不会为你工作了。
- 客户端在 DNS 请求报文中申请使用的是递归查询（也就是 RD 字段置 1 了），但在所配置的本地名称服务器上是禁用递归查询（DNS 服务器一般默认支持递归查询）的，即在应答 DNS 报文头部的 RA 字段置 0。

1. 迭代解析的基本流程

使用迭代解析方式时，如果它所配置的主名称服务器（如 Windows 系统中的“首选 DNS 服务器”）不能解析的话，客户端还会继续向所配置的其他名称服务器（如 Windows 系统中的“备用 DNS 服务器”）查询。迭代解析的基本流程如下：

（1）客户端向本机配置的本地名称服务器（在此仅以 DNS 服务器为例进行介绍，其他备用 DNS 服务器的解析流程完全一样）发出 DNS 域名查询请求。

（2）本地名称服务器收到请求后，先查询本地的缓存，如果有该域名的记录项，则本地名称服务器就直接把查询的结果返回给客户端；如果本地缓存中没有该域名的记录，则向 DNS 客户端返回一条 DNS 应答报文，报文中会给出一些参考信息，如本地名称服务器上的根名称服务器地址等。

（3）DNS 客户端在收到本地名称服务器的应答报文后，会根据其中的根名称服务器地址信息向对应的根名称服务器再次发出与前面一样的 DNS 查询请求报文。

（4）根名称服务器在收到 DNS 查询请求报文后，通过查询自己的 DNS 数据库得到请求 DNS 域名中顶级域名所对应的顶级名称服务器信息，然后以一条 DNS 应答报文返回给 DNS 客户端。

（5）DNS 客户端根据来自根名称服务器应答报文中的对应顶级名称服务器地址信息向该顶级名称服务器发出与前面一样的 DNS 查询请求报文。

（6）顶级名称服务器在收到 DNS 查询请求后，先查询自己的缓存，如果有所请求的 DNS 域名的记录项，则把对应的记录项返回给 DNS 客户端，否则通过查询后把对应域名中二级域名所对应的二级名称服务器地址信息以一条 DNS 应答报文返回给 DNS 客户端。

然后 DNS 客户端继续按照前面介绍的方法一次次地向三级、四级名称服务器查询，直到最终的权威名称服务器返回到最终的记录。

2. DNS 迭代解析示例

同样，为了方便大家理解，也举一个具体的示例，看一下 DNS 迭代解析的工作流程。本示例

也与上节中介绍的DNS递归解析一样，假设客户端想要访问自己并不识别的example.microsoft.com站点，并假设此客户端配置的本地名称服务器为dns.company.com（仅以一个为例进行介绍），在该本地名称服务器上配置的根名称服务器是a.rootserver.net。整个迭代解析过程如图11-16所示。

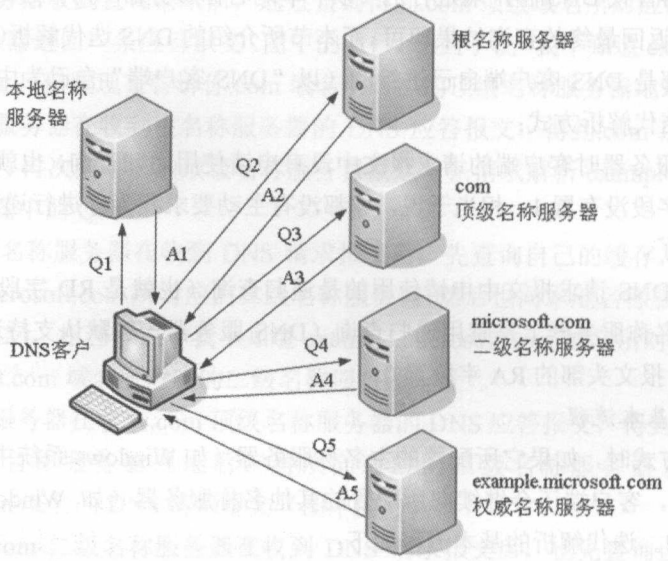


图11-16　DNS迭代解析示例

其中的Q1～Q5表示发送DNS查询请求，A1～A5是对应序号DNS查询请求的应答，具体描述如下：

（1）DNS客户端向所配置的本地名称服务器dns.company.com发出解析example.microsoft.com域名的DNS请求报文（图中的Q1）。

（2）本地名称服务器收到DNS客户端的DNS查询请求报文后，先查询本地缓存。假设没有查到该域名对应记录，则本地名称服务器把所配置的根名称服务器a.rootserver.net地址信息以DNS应答报文返回给DNS客户端（图中的A1）。

（3）DNS客户端在收到本地名称服务器的DNS应答报文后，根据其中给出的根名称服务器地址信息向对应的根名称服务器再次发送解析example.microsoft.com域名的DNS请求报文（图中的Q2）。

（4）根名称服务器在收到DNS查询请求后，通过查询得到.com顶级域名所对应的顶级名称服务器，然后把查询到的对应顶级域名信息以一条DNS应答报文返回给DNS客户端（图中的A2）。

（5）DNS客户端在收到根名称服务器的DNS应答报文，得到.com顶级域名所对应的顶级名称服务器地址后，再次向对应的顶级名称服务器发送一条解析example.microsoft.com域名的DNS请求报文（图中的Q3）。

（6）.com顶级名称服务器在收到DNS客户端的DNS查询请求报文后，先查询自己的缓存，

假设也没有该域名的记录项，则查询 microsoft.com 所对应的二级名称服务器，然后把查询到的对应二级域名信息以一条 DNS 应答报文返回给 DNS 客户端（图中的 A3）。

（7）DNS 客户端在收到.com 顶级名称服务器的 DNS 应答报文，得到 microsoft.com 二级域名所对应的二级名称服务器地址后，再次向对应的二级名称服务器发送一条解析 example.microsoft.com 域名的 DNS 请求报文（图中的 Q4）。

（8）microsoft.com 二级名称服务器在收到 DNS 客户端的 DNS 查询请求报文后，也先查询自己的缓存，假设也没有该域名的记录项，则查询 example.microsoft.com 所对应的权威名称服务器（因为这个名称服务器已包括了整个域名 example.microsoft.com 所在的区域），然后把查询到的对应权威域名信息以一条 DNS 应答报文返回给 DNS 客户端（图中的 A5）。

（9）DNS 客户端在收到 microsoft.com 二级名称服务器的 DNS 应答报文，得到 example.microsoft.com 三级域名所对应的权威名称服务器地址后，再次向对应的权威名称服务器发送解析 example.microsoft.com 域名的 DNS 请求报文（图中的 Q5）。

（10）权威名称服务器在收到 DNS 客户端的 DNS 查询请求报文后，在它的 DNS 区域数据库中查找，最终得出了 example.microsoft.com 域名所对应的 IP 地址，然后向 DNS 客户端返回一条 DNS 应答报文（图中的 A5）。这样 DNS 客户端就可以正常访问这个网站了。

如果在步骤（10）中对应域名的权威名称服务器都说找不到对应的域名记录，则会向 DNS 客户端返回一条查询失败的 DNS 应答报文。当然，如果这个权威名称服务器上配置了指向其他名称服务器的转发器，则权威名称服务器还会在转发器指向的名称服务器上进一步查询。另外，如果 DNS 客户端上配置了多个 DNS 服务器，则还会继续向其他 DNS 服务器查询。

11.4 DHCP 服务

DHCP（动态主机配置协议）服务是一种用于简化主机 IP 配置管理的服务。通过采用 DHCP 服务，可以使用 DHCP 服务器为网络上配置了 DHCP 服务客户端程序的客户端进行动态 IP 地址分配和其他相关配置，而不用管理员对各客户端主机一一进行配置，减少了许多管理任务。

11.4.1 BOOTP 和 DHCP 简介

DHCP 是早期 BOOTP（Bootstrap Protocol，自举协议）的增强版本。BOOTP 是一个基于 IP 和 UDP 的协议，最早出现在 UNIX 系统中，负责 UNIX 终端的远程启动，后来在 Windows 无盘网络中也被广泛使用。它可以让无盘站点从一个中心服务器上获得 IP 地址，为局域网中的无盘工作站分配动态 IP 地址，并且不需要每个用户去设置静态 IP 地址。

使用 BOOTP 协议的时候，一般包括“自举协议服务端”和“自举协议客户端”两部分。在为无盘站点分配 IP 地址时，首先客户端网卡会以 0.0.0.0 地址向服务器端发出 IP 地址分配请求的帧，其中就包括站点网卡的 MAC 地址。BOOTP 服务器在接收到这个请求帧后，根据该帧中的 MAC 地址在自己的启动数据库中查找这个 MAC 的记录，如果没有此 MAC 的记录则不应答这个请求；

如果有就向站点返回一条 FOUND 帧。FOUND 帧中包含的主要信息有客户端的 IP 地址、服务器的 IP 地址、硬件类型、网关 IP 地址、站点 MAC 地址和启动映象文件名。站点就根据 FOUND 帧中的信息获得分配的 IP 地址信息，然后通过 TFTP 服务器下载启动映象文件，并将此文件在工作站内存模拟成磁盘，从这个模拟磁盘启动。

BOOTP 协议有两个致使的缺点：一是在为站点分配 IP 地址前必须在服务器端事先配置好对应站点的 MAC 地址，这样一来管理员的工作量仍比较大（主要体现在事先必须收集各站点的 MAC 地址上）；二是 BOOTP 分配的 IP 地址是静态的，没有租约期的概念，一个站点分配了一个 IP 地址后即可永久使用这个 IP 地址，直到站点重启或关机，这样就形成了一个一对一的静态关系，不利于 IP 地址的有效使用。

DHCP 是 BOOTP 的增强版本，也是基于 IP 和 UDP 协议的。它全面克服了 BOOTP 的以上不足：一方面在 DHCP 服务器端根本不用配置客户端的 MAC 地址，这样也就不需要管理员去收集各客户端的 MAC 地址；另一方面 DHCP 通过“租约”的技术可以大大提高 IP 地址的使用效率和安全性，不再是一对一的静态关系，是一种动态分配关系，同时在功能上 DHCP 相对 BOOTP 更有全面的增强，通过 DHCP 中继代理功能，一个 DHCP 服务器还可以为多个网段的客户端自动分配 IP 地址。

DHCP 服务的目的是旨在通过使用配置了 DHCP 服务的计算机或网络设备（如三层交换机、路由器，甚至是防火墙）集中管理网络上使用的 IP 地址和其他相关配置来降低管理地址配置的复杂性。如在 Windows 2000 Server/ Server 2003/Server 2008 家族和各种 Linux 和 UNIX 服务器操作系统中都提供了 DHCP 服务，特别是各种三层交换机、路由器（包括宽带路由器）等都全面支持 DHCP 服务。最新的 DHCP 草案标准为 RFC 2131。

DHCP 服务之所以能为 DHCP 客户端自动分配 IP 地址，根本原因是在 DHCP 服务器中已准备好了用来为客户端分配 IP 地址的 IP 地址池。这个 IP 地址池就像装满了可用于分配的许多 IP 地址，而且这些 IP 地址是属于一个网段的一部分或者全部的 IP 地址。有关 DHCP 的 IP 地址分配原理将在后面小节中介绍。

11.4.2 DHCP 服务的主要功能及应用环境

通过上节我们简单地了解了 DHCP 服务的一些基础知识，本节综合介绍一下 DHCP 服务的主要功能、使用 DHCP 服务的好处和 DHCP 服务的主要应用环境。

1. DHCP 服务的主要功能

DHCP 服务不仅是简单的自动 IP 地址分配，它还可以提供以下附加功能：

- 通过 IP 地址与 MAC 地址绑定功能实现静态 IP 地址分配。
- 配置客户端的 DNS 服务器、WINS 服务器（仅限微软 Windows 系统中的 DHCP 服务器）和默认网关。
- IP 地址排除，使已静态分配给其他主机（特别是各种服务器）的 IP 地址不再分配给其他 DHCP 客户端。

- 通过 DHCP 中继功能，一个 DHCP 服务器可以为多个网段（或 VLAN）中的 DHCP 客户端分配不同地址池中的 IP 地址，进一步简化网络中的 IP 地址配置工作。

2. 使用 DHCP 服务的好处

DHCP 服务为管理基于 TCP/IP 的网络提供了以下好处：

（1）减少 TCP/IP 协议配置和管理的工作量。

因为使用了 DHCP 服务后，安装了 DHCP 客户程序（现在所有操作系统都自带并安装了这一程序）的客户端计算机可直接选择像图 11-17 所示那样的“自动获得 IP 地址”和“自动获得 DNS 服务器地址”选项，这样客户端就无需配置 IP 地址、子网掩码、网关、DNS 服务器地址这类 TCP/IP 信息。

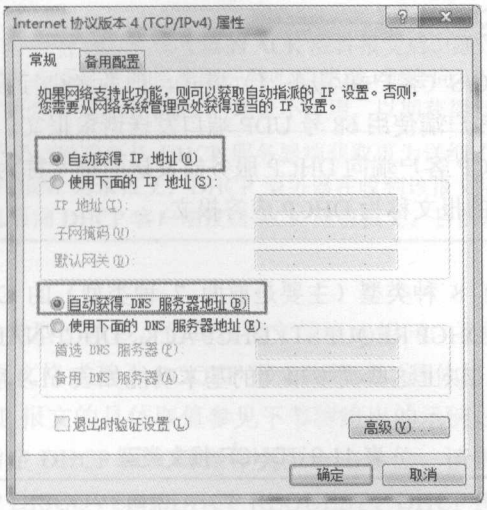


图 11-17　DHCP 选项配置对话框

由此可见，DHCP 服务可大大减轻管理员的工作负担（因为一般 TCP/IP 协议的配置是需要管理员权限的）。但是由 DHCP 服务器自动分配的 IP 地址都有一个租约期，也就是自分配给某个客户端开始，该客户端只能在这个租约期内使用所分配的这个 IP 地址，过期后可以续约或者重新申请。

（2）方便客户端移动。

使用 DHCP 服务自动为客户端分配 IP 地址还有一个好处就是方便客户端的移动，这对于需要移动办公的员工来说非常实用。因为这样客户端连接在哪里就可以从对应网段（或 VLAN）配置的 DHCP 服务器上自动分配对应网段的 IP 地址。反过来，如果你采用的是静态 IP 地址分配方式，则客户端每次移动到一个其他网段（或 VLAN）时，管理员都需要重新设置一次 TCP/IP 信息。

（3）配置更加可靠。

采用 DHCP 服务自动分配 IP 地址信息就可以很有效地避免由于配置时键入错误而引起的配置错误，还有助于防止由于在网络上配置新的计算机时因重用以前指派的 IP 地址而引起的地址冲突。

3. DHCP 服务的主要应用环境

在以下场合通常利用 DHCP 服务来完成 IP 地址分配：

- 网络规模较大，手工配置需要很大的工作量，并难以对整个网络进行集中管理。当然像各种服务器、网络设备节点都是需要采用静态 IP 地址分配的，否则用户可能无法访问你的服务器，网络设备也无法进行正常的数据转发和路由功能。
- 网络中主机数目大于该网络支持的IP地址数量，无法给每台主机分配一个固定的IP地址。例如，Internet 接入服务提供商限制同时接入网络的用户数目，大量用户必须动态获得自己的 IP 地址。
- 网络中只有少数主机需要固定的 IP 地址，大多数主机没有固定的 IP 地址需求。

11.4.3 DHCP 报文及其格式

DHCP 服务一样工作在 C/S（客户端/服务器）模式，但两者进行报文传输时所使用的 UDP 传输端口是不一样的，DHCP 客户端使用 68 号 UDP 端口发送请求报文，而 DHCP 服务器使用 67 号 UDP 端口发送应答报文。DHCP 客户端向 DHCP 服务器发送的报文称为 DHCP 请求报文，而 DHCP 服务器向 DHCP 客户端发送的报文称为 DHCP 应答报文。

1. DHCP 报文种类

整个 DHCP 服务一共有 8 种类型（主要是前面 7 种类型）的 DHCP 报文，分别为 DHCP DISCOVER、DHCP OFFER、DHCP REQUEST、DHCP ACK、DHCP NAK、DHCP RELEASE、DHCP DECLINE、DHCP INFORM。以上这些类型报文的基本功能如表 11-9 所示。

表 11-9 DHCP 报文类型

DHCP 报文类型	说明
DHCP DISCOVER	因为 DHCP 客户端在请求 IP 地址时并不知道 DHCP 服务器的位置，所以 DHCP 客户端会在本地网络内以广播方式发送 DISCOVER 请求报文，以发现网络中的 DHCP 服务器。所有收到 DISCOVER 报文的 DHCP 服务器都会发送应答报文，DHCP 客户端据此可以知道网络中存在的 DHCP 服务器的位置
DHCP OFFER	DHCP 服务器收到 DISCOVER 报文后，就会在所配置的地址池中查找一个合适的 IP 地址，加上相应的租约期限和其他配置信息（如网关、DNS 服务器等），构造一个 OFFER 报文，发送给 DHCP 客户端，告知用户本服务器可以为其提供 IP 地址。但这个报文只是告诉 DHCP 客户端可以提供 IP 地址，最终还需要客户端通过 ARP 来检测该 IP 地址是否重复
DHCP REQUEST	DHCP 客户端可能会收到很多 OFFER 请求报文，所以必须在这些应答中选择一个。通常是选择第一个 OFFER 应答报文的服务器作为自己的目标服务器，并向该服务器发送一个广播的 REQUEST 请求报文，通告选择的服务器希望获得所分配的 IP 地址。另外，DHCP 客户端在成功获取 IP 地址后，在地址使用租期过去 1/2 时也会向 DHCP 服务器发送单播 REQUEST 请求报文请求续延租约，如果没有收到 ACK 报文，在租期过去 3/4 时会再次发送广播的 REQUEST 请求报文以请求续延租约

续表

DHCP 报文类型	说明
DHCP ACK	DHCP 服务器收到 REQUEST 请求报文后，根据 REQUEST 报文中携带的用户 MAC 地址来查找有没有相应的租约记录，如果有则发送 ACK 应答报文，通知用户可以使用分配的 IP 地址
DHCP NAK	如果 DHCP 服务器收到 REQUEST 请求报文后，没有发现有相应的租约记录或者由于某些原因无法正常分配 IP 地址，则向 DHCP 客户端发送 NAK 应答报文，通知用户无法分配合适的 IP 地址
DHCP RELEASE	当 DHCP 客户端不再需要使用分配的 IP 地址时，就会主动向 DHCP 服务器发送 RELEASE 请求报文，告知服务器用户不再需要分配 IP 地址，请求 DHCP 服务器释放对应的 IP 地址
DHCP DECLINE	DHCP 客户端收到 DHCP 服务器的 ACK 应答报文后，通过地址冲突检测发现服务器分配的地址冲突或者由于其他原因导致不能使用，则会向 DHCP 服务器发送 DECLINE 请求报文，通知服务器所分配的 IP 地址不可用，以期获得新的 IP 地址
DHCP INFORM	DHCP 客户端如果需要从 DHCP 服务器端获取更为详细的配置信息，则向 DHCP 服务器发送 INFORM 请求报文；DHCP 服务器在收到该报文后，将根据租约在查找到相应的配置信息后向 DHCP 客户端发送 ACK 应答报文。目前基本上不用了

2. DHCP 报文格式

虽然 DHCP 服务的报文类型比较多，但每种报文的格式相同，不同类型的报文只是报文中的某些字段取值不同。DHCP 报文格式基于 BOOTP 报文的格式，具体格式如图 11-18 所示。下面是各字段的说明，至于各 DHCP 报文的具体取值参见下节将给出的示例。

- OP：Operation，指定 DHCP 报文的操作类型，占 8 位。请求报文置 1，应答报文置 2，表 11-9 中的 DHCP DISCOVER、DHCP REQUEST、DHCP RELEASE、DHCP INFORM 和 DHCP DECLINE 为请求报文，而 DHCP OFFER、DHCP ACK 和 DHCP NAK 为应答报文。
- Htype、Hlen：分别指定 DHCP 客户端的 MAC 地址类型及长度，各占 8 位。MAC 地址类型其实是指明网络类型，Htype 字段置 1 时表示为最常见的以太网 MAC 地址类型；以太网 MAC 地址长度为 6 个字节，即对应 Hlen 字段值为 6。
- Hops：DHCP 报文经过的 DHCP 中继的数目，占 8 位。DHCP 请求报文每经过一个 DHCP 中继，该字段就会增加 1。没有经过 DHCP 中继时值为 0。
- Xid：客户端通过 DHCP DISCOVER 报文发起一次 IP 地址请求时选择的随机数，相当于请求标识，占 32 位，用来标识一次 IP 地址请求过程。在一次请求中所有报文的 Xid 都是一样的。
- Secs：DHCP 客户端从获取到 IP 地址或者续约过程开始到现在所消耗的时间，以秒为单位，占 16 位。在没有获得 IP 地址前该字段始终为 0。

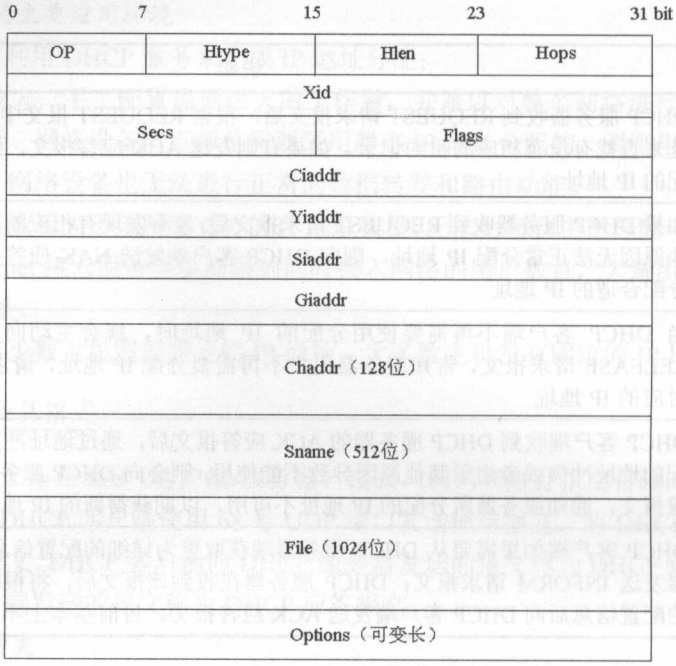


图 11-18　DHCP 报文格式

- Flags：标志位，占 16 位，第一位为广播应答标识位，用来标识 DHCP 服务器应答报文是采用单播方式还是广播方式发送，置 0 时表示采用单播发送方式，置 1 时表示采用广播发送方式。其余位保留不用。

【注意】在客户端正式分配了 IP 地址之前的第一次 IP 地址请求过程中，所有 DHCP 报文都是以广播方式发送的，包括客户端发送的 DHCP DISCOVER 和 DHCP REQUEST 报文，以及 DHCP 服务器发送的 DHCP OFFER、DHCP ACK 和 DHCP NAK 报文。当然，如果是由 DHCP 中继器转的报文，则都是以单播方式发送的，具体将在 11.4.6 节中体现。另外，IP 地址续约、IP 地址释放的相关报文都是采用单播方式进行发送的，具体将在 11.4.3 节中体现。

- Ciaddr：指示 DHCP 客户端的 IP 地址，占 32 位（4 个字节）。仅在 DHCP 服务器发送的 ACK 报文中显示，在其他报文中均显示 0，因为在得到 DHCP 服务器确认前 DHCP 客户端是还没有分配到 IP 地址的。
- Yiaddr：指示 DHCP 服务器分配给客户端的 IP 地址，占 32 位（4 个字节）。仅在 DHCP 服务器发送的 OFFER 和 ACK 报文中显示，其他报文中显示为 0。
- Siaddr：指示下一个为 DHCP 客户端分配 IP 地址等信息的 DHCP 服务器 IP 地址，占 32 位（4 个字节）。仅在 DHCP OFFER、DHCP ACK 报文中显示，其他报文中显示为 0。
- Giaddr：指示 DHCP 客户端发出请求报文后经过的第一个 DHCP 中继的 IP 地址，占 32 位（4 个字节）。如果没有经过 DHCP 中继，则显示为 0。

- Chaddr：指示 DHCP 客户端的 MAC 地址，占 128 位（16 个字节），以太网中只用到低 48 位。在每个报文中都会显示对应 DHCP 客户端的 MAC 地址。
- Sname：指示为 DHCP 客户端分配 IP 地址的 DHCP 服务器名称（DNS 域名格式），占 512 位（64 个字节）。在 OFFER 和 ACK 报文中显示发送报文的 DHCP 服务器名称，其他报文显示为空。
- File：指示 DHCP 服务器为 DHCP 客户端指定的启动配置文件名称及路径信息，占 1024 位（128 个字节）。仅在 DHCP OFFER 报文中显示，其他报文中显示为空。
- Option：可选项字段，长度可变。在此部分可选的选项包含：报文类型（代码为 53，占 1 个字节）、有效租约期（代码为 51，以秒为单位，占 4 个字节）、续约时间（代码为 58，占 4 个字节）、子网掩码（代码为 1，占 4 个字节）、默认网关（代码为 3，可以是一个路由器 IP 地址列表，长度可变，但必须是 4 个字节的倍数）、DNS 服务器（代码为 6，可以是一个 DNS 服务器 IP 地址列表，长度可变，但必须是 4 个字节的整数倍）、域名称（代码为 15，主 DNS 服务器名称，长度可变）、WINS 服务器（代码为 44，可以是一个 WINS 服务器 IP 地址列表，长度可变，但必须是 4 个字节的倍数）等配置信息。表 11-9 所示的 DHCP 报文类型的取值分别如下：
  - DHCP DISCOVER：1。
  - DHCP OFFER：2。
  - DHCP REQUEST：3。
  - DHCP DECLINE：4。
  - DHCP ACK：5。
  - DHCP NAK：6。
  - DHCP RELEASE：7。

11.4.4 DHCP 服务 IP 地址自动分配原理

DHCP 在提供服务时，DHCP 客户端是以 UDP 68 号端口进行数据传输的，而 DHCP 服务器是以 UDP 67 号端口进行数据传输的。DHCP 服务不仅体现在为 DHCP 客户端提供 IP 地址自动分配的过程中，还体现在后面的 IP 地址续约和释放过程中。本节仅介绍 DHCP 客户端初次分配 IP 地址的过程。

在整个 DHCP 服务器为 DHCP 客户端初次提供 IP 地址自动分配的过程中，一共经过了以下 4 个阶段，利用了表 11-9 中的前 4 个报文：发现阶段（DHCP 客户端在网络中广播发送 DHCP DISCOVER 请求报文，发现 DHCP 服务器，请求 IP 地址租约）、提供阶段（DHCP 服务器通过 DHCP OFFER 报文向 DHCP 客户端提供 IP 地址预分配）、选择阶段（DHCP 客户端通过 DHCP REQUEST 报文确认选择第一个 DHCP 服务器为它提供 IP 地址自动分配服务）和确认阶段（被选择的 DHCP 服务器通过 DHCP ACK 报文把在 DHCP OFFER 报文中准备的 IP 地址租约给对应 DHCP 客户端）。

在 DHCP 客户端，在获得了一个 IP 地址以后就可以发送一个 ARP 请求来避免由于 DHCP 服

务器地址池重叠而引发的 IP 冲突。以上 4 个阶段如图 11-19 所示，相当于 DHCP 客户端与 DHCP 服务器的四次握手过程。

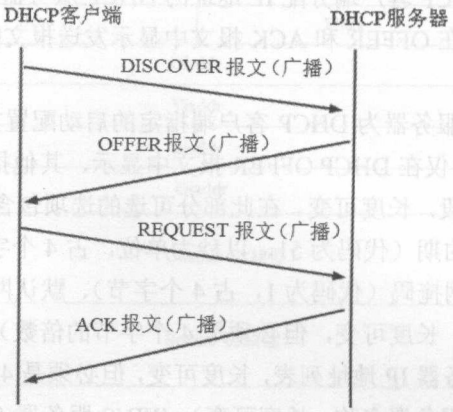


图 11-19　DHCP 客户端从 DHCP 服务器获取 IP 地址的 4 个阶段

具体描述如下：

（1）发现阶段，即 DHCP 客户端获取网络中 DHCP 服务器信息的阶段。在客户端配置了 DHCP 客户端程序并启动后，以广播方式发送 DHCP DISCOVER 报文寻找网络中的 DHCP 服务器。此广播报文采用传输层的 UDP 68 号端口发送（封装的目的端口为 UDP 68 号端口），经过网络层 IP 协议封装后，源 IP 地址为 0.0.0.0（因为此时还没有分配 IP 地址），目的 IP 地址为 255.255.255.255（有限广播 IP 地址）。

如下是一个 DHCP DISCOVER 报文封装的 IP 报头示例，可以看到 Destination Address（目的地址）是 255.255.255.255，而 Source Address（源地址）是 0.0.0.0。IP 报头中的其他字段含义大家可以参见 8.3.4 节中的相关内容。

```
IP:ID = 0x0; Proto = UDP; Len: 328
IP:Version = 4 (0x4)
IP:Header Length = 20 (0x14)
IP:Service Type = 0 (0x0)
IP:Precedence = Routine
IP:...0.... = Normal Delay
IP:....0... = Normal Throughput
IP:.....0.. = Normal Reliability
IP:Total Length = 328 (0x148)
IP:Identification = 0 (0x0)
IP:Flags Summary = 0 (0x0)
IP:.......0 = Last fragment in datagram
IP:.......0. = May fragment datagram if necessary
IP:Fragment Offset = 0 (0x0) bytes
IP:Time to Live = 128 (0x80)
IP:Protocol = UDP - User Datagram   !---使用 UDP 传输层协议
IP:Checksum = 0x39A6
```

```
IP:Source Address = 0.0.0.0      !---源 IP 地址为 0.0.0.0
IP:Destination Address = 255.255.255.255      !----目的 IP 地址为 255.255.255.255
IP:Data:Number of data bytes remaining = 308 (0x0134)
```



在以上 DHCP DISCOVER 报文中 IP 报头中的目的地址(Destination Address)是 255.255.255.255 这个有限广播地址。这个有限广播地址就是代表任意一个 IPv4 子网的广播地址，当然是发送报文的主机所在的子网和 DHCP 服务器所在子网的广播地址，但因为此时 DHCP 客户端并不知道 DHCP 服务器所在的是哪个子网。下面所有其他 DHCP 报文中的 255.255.255.255 地址的含义也是一样的。

至于 IP 报头中的源地址 (Source Address)，由于当前 DHCP 客户端主机并未分配具体的 IP 地址，所以只能用具有任意代表功能的 0.0.0.0 地址来表示了。下面所有其他 DHCP 报文中指定的 0.0.0.0 地址的含义也是一样的。

因为此时，DHCP 客户端没有分配到 IP 地址，也不知道 DHCP 服务器或 DHCP 中继的 IP 地址，所以在 DHCP DISCOVER 报文中 Ciaddr（客户端 IP 地址）、Yiaddr（被分配的 DHCP 客户端 IP 地址）、Siaddr（下一个为 DHCP 客户端分配 IP 地址的 DHCP 服务器地址）、Giaddr（DHCP 中继 IP 地址）这 4 个字段均为 0.0.0.0。另外，从中可以看到，在 Chaddr 字段和 DHCP 选项中的 Client Identifier 字段中都标识了 DHCP 客户端网卡的 MAC 地址。

```
DHCP:Discover                (xid=21274A1D)
DHCP:Op Code                 (op)       = 1 (0x1)
DHCP:Hardware Type           (htype)    = 1 (0x1) 10Mb Ethernet
DHCP:Hardware Address Length (hlen) = 6 (0x6)
DHCP:Hops                    (hops)     = 0 (0x0)
DHCP:Transaction ID      (xid)      = 556223005 (0x21274A1D)
DHCP:Seconds                 (secs)     = 0 (0x0)
DHCP:Flags                   (flags)    = 1 (0x1)     !---标志位置 1，代表以广播方式发送
DHCP:1..... = Broadcast
DHCP:Client IP Address (ciaddr) = 0.0.0.0
DHCP:Your    IP Address (yiaddr) = 0.0.0.0
DHCP:Server IP Address (siaddr) = 0.0.0.0
DHCP:Relay   IP Address (giaddr) = 0.0.0.0
DHCP:Client Ethernet Address (chaddr) = 08002B2ED85E
DHCP:Server Host Name    (sname)    = <Blank>
DHCP:Boot File Name      (file)     = <Blank>
DHCP:Magic Cookie = [OK]
DHCP:Option Field        (options)
DHCP:DHCP Message Type               = DHCP Discover     !---DHCP 报文类型为 DHCP DISCOVER
DHCP:Client-identifier               = (Type:1) 08 00 2b 2e d8 5e
DHCP:Host Name                       = JUMBO-WS      !---DHCP 客户端的主机名
DHCP:Parameter Request List = (Length:7) 01 0f 03 2c 2e 2f 06
DHCP:End of this option field
```

（2）提供阶段，即 DHCP 服务器向 DHCP 客户端提供预分配 IP 地址的阶段。网络中的所有 DHCP 服务器接收到客户端的 DHCP DISCOVER 报文后都会根据自己地址池中 IP 地址分配的优先

次序选出一个 IP 地址，然后与其他参数一起通过传输层的 UDP 67 号端口在 DHCP OFFER 报文中以广播方式发送给客户端（目的端口是 DHCP 客户端的 UDP 68 号端口）。客户端通过封装在帧中的目的 MAC 地址（也就是 DHCP DISCOVER 报文中的 CHADDR 字段值）的比对来确定是否接收该帧。但这样一来，理论上 DHCP 客户端可能会收到多个 DHCP OFFER 报文（当网络中存在多个 DHCP 服务器时），但 DHCP 客户端只接收第一个到来的 DHCP OFFER 报文。

DHCP OFFER 报文经过 IP 协议封装后的源 IP 地址是 DHCP 服务器自己的 IP 地址，目的地址仍是 255.255.255.255 广播地址，使用的协议仍为 UDP。下面是一个 DHCP OFFER 报文的 IP 报头示例。

```
IP:ID = 0x3C30; Proto = UDP; Len: 328
IP:Version = 4 (0x4)
IP:Header Length = 20 (0x14)
IP:Service Type = 0 (0x0)
IP:Precedence = Routine
IP:...0.... = Normal Delay
IP:....0... = Normal Throughput
IP:.....0.. = Normal Reliability
IP:Total Length = 328 (0x148)
IP:Identification = 15408 (0x3C30)
IP:Flags Summary = 0 (0x0)
IP:.....0 = Last fragment in datagram
IP:.....0. = May fragment datagram if necessary
IP:Fragment Offset = 0 (0x0) bytes
IP:Time to Live = 128 (0x80)
IP:Protocol = UDP - User Datagram
IP:Checksum = 0x2FA8
IP:Source Address = 157.54.48.151
IP:Destination Address = 255.255.255.255
IP:Data:Number of data bytes remaining = 308 (0x0134)
```

在 DHCP OFFER 报文中，Ciaddr 字段值仍为 0.0.0.0，因为客户端仍没有分配到 IP 地址；Yiaddr 字段已有值了，这是 DHCP 服务器为该客户端预分配的 IP 地址（但会带上 DHCP 服务器的 IP 地址 Siaddr）；因为没有经过 DHCP 中继服务器，所以 Giaddr 字段值仍为 0.0.0.0。另外，在 DHCP 可选项部分，可以看到由服务器随 IP 地址一起发送的各种选项。在这种情况下，服务器发送的是子网掩码、默认网关（路由器）、租约时间、WINS 服务器地址（NetBIOS 名称服务）和 NetBIOS 节点类型。下面是一个 DHCP OFFER 报文的示例。

```
DHCP:Offer                    (xid=21274A1D)
DHCP:Op Code                  (op)       = 2 (0x2)
DHCP:Hardware Type            (htype)    = 1 (0x1) 10Mb Ethernet
DHCP:Hardware Address Length (hlen) = 6 (0x6)
DHCP:Hops                     (hops)     = 0 (0x0)
DHCP:Transaction ID           (xid)      = 556223005 (0x21274A1D)
DHCP:Seconds                  (secs)     = 0 (0x0)
DHCP:Flags                    (flags)    = 1 (0x1)
DHCP:1..... =  Broadcast
DHCP:Client IP Address (ciaddr) = 0.0.0.0
```

```
DHCP:Your    IP Address (yiaddr) = 157.54.50.5
DHCP:Server IP Address (siaddr) = 157.54.48.151
DHCP:Relay   IP Address (giaddr) = 0.0.0.0
DHCP:Client Ethernet Address (chaddr) = 08002B2ED85E
DHCP:Server Host Name   (sname)  = Server1      !---DHCP 服务器主机名
DHCP:Boot File Name     (file)   = <Blank>
DHCP:Magic Cookie = [OK]
DHCP:Option Field       (options)
DHCP:DHCP Message Type        = DHCP Offer          !---DHCP 报文类型为 DHCP OFFER
DHCP:Subnet Mask              = 255.255.240.0       !---所分配 IP 地址的子网掩码为 255.255.240.0
DHCP:Renewal Time Value (T1) = 8 Days,  0:00:00     !---想要继续租约原来分配的 IP 地址，则提出续约申请的
                                                    !---期限为 8 天
DHCP:Rebinding Time Value (T2) = 14 Days,  0:00:00  !---如果上次申请续约失败，再次申请绑定原来分配到的
                                                    !---IP 地址的期限为 14 天
DHCP:IP Address Lease Time   = 16 Days,  0:00:00    !---租约期限为 16 天，也就是 DHCP 客户端可使用此 IP
                                                    !---地址的最长时间为 16 天
DHCP:Server Identifier        = 157.54.48.151       !---DHCP 服务器的 IP 地址为 157.54.48.151
DHCP:Router                   = 157.54.48.1         !---默认网关 IP 地址为 157.54.48.1
DHCP:NetBIOS Name Service     = 157.54.16.154       !---DNS 服务器 IP 地址为 157.54.16.154
DHCP:NetBIOS Node Type        = (Length: 1) 04
DHCP:End of this option field
```

（3）选择阶段，即 DHCP 客户端选择 IP 地址的阶段。如果有多台 DHCP 服务器向该客户端发来 DHCP OFFER 报文，客户端只接收第一个收到的 DHCP OFFER 报文，然后以广播方式发送 DHCP REQUEST 报文。在该报文的 Requested Address 选项中包含 DHCP 服务器在 DHCP OFFER 报文中预分配的 IP 地址、对应的 DHCP 服务器 IP 地址等。这样也就相当于同时告诉其他 DHCP 服务器，它们可以释放已提供的地址并将这些地址返回到可用的地址池中。

在 DHCP REQUEST 报文封装的 IP 协议头部中，客户端的 Source Address 仍然是 0.0.0.0，数据包的 Destination 仍然是 255.255.255.255。但在 DHCP REQUEST 报文中 Ciaddr、Yiaddr、Siaddr、Giaddr 字段的地址均为 0.0.0.0，大家自己分析一下这是为什么，其实很简单的。下面是一个 DHCP REQUEST 报文头部和 DHCP REQUEST 报文的示例。

```
IP:ID = 0x100; Proto = UDP; Len: 328
IP:Version = 4 (0x4)
IP:Header Length = 20 (0x14)
IP:Service Type = 0 (0x0)
IP:Precedence = Routine
IP:...0.... = Normal Delay
IP:....0... = Normal Throughput
IP:.....0.. = Normal Reliability
IP:Total Length = 328 (0x148)
IP:Identification = 256 (0x100)
IP:Flags Summary = 0 (0x0)
IP:.....0 = Last fragment in datagram
IP:.....0. = May fragment datagram if necessary
IP:Fragment Offset = 0 (0x0) bytes
IP:Time to Live = 128 (0x80)
IP:Protocol = UDP - User Datagram
```

```
IP:Checksum = 0x38A6
IP:Source Address = 0.0.0.0
IP:Destination Address = 255.255.255.255
IP:Data:Number of data bytes remaining = 308 (0x0134)

DHCP:Request                    (xid=21274A1D)
DHCP:Op Code                    (op)        = 1 (0x1)
DHCP:Hardware Type              (htype)     = 1 (0x1) 10Mb Ethernet
DHCP:Hardware Address Length (hlen) = 6 (0x6)
DHCP:Hops                       (hops)      = 0 (0x0)
DHCP:Transaction ID       (xid)       = 556223005 (0x21274A1D)
DHCP:Seconds                    (secs)      = 0 (0x0)
DHCP:Flags                      (flags)     = 1 (0x1)
DHCP:1..... = Broadcast
DHCP:Client IP Address (ciaddr) = 0.0.0.0
DHCP:Your     IP Address (yiaddr) = 0.0.0.0
DHCP:Server IP Address (siaddr) = 0.0.0.0
DHCP:Relay    IP Address (giaddr) = 0.0.0.0
DHCP:Client Ethernet Address (chaddr) = 08002B2ED85E
DHCP:Server Host Name    (sname)    = Server1
DHCP:Boot File Name      (file)     = <Blank>
DHCP:Magic Cookie = [OK]
DHCP:Option Field          (options)
DHCP:DHCP Message Type          = DHCP Request
DHCP:Client-identifier          = (Type:1) 08 00 2b 2e d8 5e
DHCP:Requested Address          = 157.54.50.5
DHCP:Server Identifier          = 157.54.48.151
DHCP:Host Name                  = JUMBO-WS
DHCP:Parameter Request List = (Length:7) 01 0f 03 2c 2e 2f 06
DHCP:End of this option field
```

（4）确认阶段，即 DHCP 服务器确认分配级 DHCP 客户端 IP 地址的阶段。某个 DHCP 服务器在收到 DHCP 客户端发来的 DHCP REQUEST 报文后，只有 DHCP 客户端选择的服务器会进行如下操作：如果确认将地址分配给该客户端，则以广播方式返回 DHCP ACK 报文；否则返回 DHCP NAK 报文，表明地址不能分配给该客户端。

在 DHCP 服务器发送的 DHCP ACK 报文的 IP 协议头部，Source Address 是 DHCP 服务器 IP 地址，Destination Address 仍然是广播地址 255.255.255.255。在 DHCP ACK 报文中 Yiaddr 字段包含要分配给客户端的 IP 地址，而 Chaddr 和 DHCP:Client Identifier 字段是发出请求的客户端中网卡的 MAC 地址。同时，在选项部分也会把在 DHCP OFFER 报文中所分配的 IP 地址的子网掩码、默认网关、DNS 服务器、租约期、续约时间等信息加上。

```
IP:ID = 0x3D30; Proto = UDP; Len: 328
IP:Version = 4 (0x4)
IP:Header Length = 20 (0x14)
IP:Service Type = 0 (0x0)
IP:Precedence = Routine
IP:...0.... = Normal Delay
IP:.....0... = Normal Throughput
```

```
IP:.....0.. = Normal Reliability
IP:Total Length = 328 (0x148)
IP:Identification = 15664 (0x3D30)
IP:Flags Summary = 0 (0x0)
IP:.....0 = Last fragment in datagram
IP:.....0. = May fragment datagram if necessary
IP:Fragment Offset = 0 (0x0) bytes
IP:Time to Live = 128 (0x80)
IP:Protocol = UDP - User Datagram
IP:Checksum = 0x2EA8
IP:Source Address = 157.54.48.151
IP:Destination Address = 255.255.255.255
IP:Data:Number of data bytes remaining = 308 (0x0134)

DHCP:ACK                    (xid=21274A1D)
DHCP:Op Code          (op)      = 2 (0x2)
DHCP:Hardware Type    (htype)   = 1 (0x1) 10Mb Ethernet
DHCP:Hardware Address Length (hlen) = 6 (0x6)
DHCP:Hops             (hops)    = 0 (0x0)
DHCP:Transaction ID   (xid)     = 556223005 (0x21274A1D)
DHCP:Seconds          (secs)    = 0 (0x0)
DHCP:Flags            (flags)   = 1 (0x1)
DHCP:1..... = Broadcast
DHCP:Client IP Address (ciaddr) = 0.0.0.0
DHCP:Your    IP Address (yiaddr) = 157.54.50.5
DHCP:Server IP Address (siaddr) = 157.54.48.151
DHCP:Relay   IP Address (giaddr) = 0.0.0.0
DHCP:Client Ethernet Address (chaddr) = 08002B2ED85E
DHCP:Server Host Name  (sname)  = Server1
DHCP:Boot File Name    (file)   = <Blank>
DHCP:Magic Cookie = [OK]
DHCP:Option Field      (options)
DHCP:DHCP Message Type        = DHCP ACK
DHCP:Renewal Time Value (T1) = 8 Days,  0:00:00
DHCP:Rebinding Time Value (T2) = 14 Days,  0:00:00
DHCP:IP Address Lease Time  = 16 Days,  0:00:00
DHCP:Server Identifier      = 157.54.48.151
DHCP:Subnet Mask            = 255.255.240.0
DHCP:Router                 = 157.54.48.1
DHCP:NetBIOS Name Service   = 157.54.16.154
DHCP:NetBIOS Node Type      = (Length: 1) 04
DHCP:End of this option field
```

【说明】客户端在收到服务器返回的 DHCP-ACK 确认报文后，会以广播的方式发送免费 ARP 报文（该报文中，源 IP 地址和目的 IP 地址都是本机 IP 地址，源 MAC 地址是本机 MAC 地址，目的 MAC 地址是广播 MAC 地址），探测是否有主机使用服务器分配的 IP 地址，如果在规定的时间内没有收到回应，客户端才使用此地址；否则，客户端会发送 DHCP DECLINE 报文给 DHCP 服务器并重新申请 IP 地址。

如果网络中存在多个 DHCP 服务器，除 DHCP 客户端选中的服务器外，其他 DHCP 服务器中本次未分配出的 IP 地址仍可分配给其他客户端。

11.4.5 DHCP 服务 IP 地址租约更新原理

如果采用动态 IP 地址分配策略，则 DHCP 服务器分配给客户端的 IP 地址都是有一定租约期限的，当租约期满后 DHCP 服务器又会收回原来分配的这个 IP 地址。如果 DHCP 客户端希望继续使用该地址，则需要向 DHCP 服务器提出更新 IP 地址租约的申请，也就是前面所说到的“续约”。IP 地址租约更新或者 IP 地址续约也就是更新服务器端对 IP 地址的租约信息，使其恢复为初始状态。申请续约的方法如下：

（1）在 DHCP 客户端的 IP 地址租约期限达到 1/2 时，由 DHCP 客户端向为它分配 IP 地址的 DHCP 服务器以单播方式发送 DHCP REQUEST 请求报文，以期进行 IP 租约的更新。

（2）如果 DHCP 服务器同意续约，则 DHCP 服务器向客户端以单播方式返回 DHCP ACK 报文，通知 DHCP 客户端已经获得新 IP 租约，可以继续使用此 IP 地址；相反，如果 DHCP 服务器不同意续约，则 DHCP 服务器以单播方式返回 DHCP NAK 报文，通知 DHCP 客户端不能获得新的租约，此 IP 地址不可以再分配给该客户端。

（3）如果上面的续约申请失败，则 DHCP 客户端还会在租约期限达到 7/8 时再次以广播方式发送 DHCP REQUEST 请求报文进行续约。DHCP 服务器的处理方式同上，不再赘述。

如果第二次续约请求还是失败，则原来租约的 IP 地址将被释放。

11.4.6 DHCP 中继代理服务

在 11.4.4 节中我们已一再强调，在 DHCP 客户端初次从 DHCP 服务器获取 IP 地址的过程中，所有从 DHCP 客户端发出的请求报文和所有由 DHCP 服务器返回的应答报文均是以广播方式（目的地址为 255.255.255.255）进行发送的，所以 DHCP 服务只适用于 DHCP 客户端和 DHCP 服务器处于同一个子网（也就是 DHCP 服务器至少有一个端口与 DHCP 客户端所在的子网是直接连接的）的情况，因为广播包是不能穿越子网的。

基于 DHCP 服务的以上限制，这样一来，如果 DHCP 客户端与 DHCP 服务器之间隔了路由器设备，不在同一子网，就不能直接通过这台 DHCP 服务器获取 IP 地址，即使 DHCP 服务器上已配置了对应的地址池。这也就意味着，如果想要让多个子网中的主机进行动态 IP 地址分配，就需要在网络中的所有子网中都设置一个 DHCP 服务器。这显然是很不经济的，也是没有必要的。

幸好，DHCP 中继功能可以很好地解决 DHCP 服务的以上难题。通过 DHCP 中继代理服务，与 DHCP 服务器不在同一子网的 DHCP 客户端可以通过 DHCP 中继代理（通常也是由路由器或三层交换机设备担当，但需要开启 DHCP 中继功能）与位于其他网段的 DHCP 服务器通信，最终使 DHCP 客户端获取到从 DHCP 服务器上分配而来的 IP 地址。此时的 DHCP 中继代理就位于 DHCP 客户端和 DHCP 服务器之间，负责广播 DHCP 报文的转发，如图 11-20 所示。很显然，DHCP 中继代理必须有一个端口连接了 DHCP 客户端所在的子网，另一端又直接连接了 DHCP 服务器所在

的子网。当然一个 DHCP 中继代理可以同时连接多个子网，作为多个 DHCP 客户端子网的中继代理。这样，多个子网上的 DHCP 客户端又可以使用同一个 DHCP 服务器来进行 IP 地址的自动分配了，既节省了成本，又便于进行集中管理。

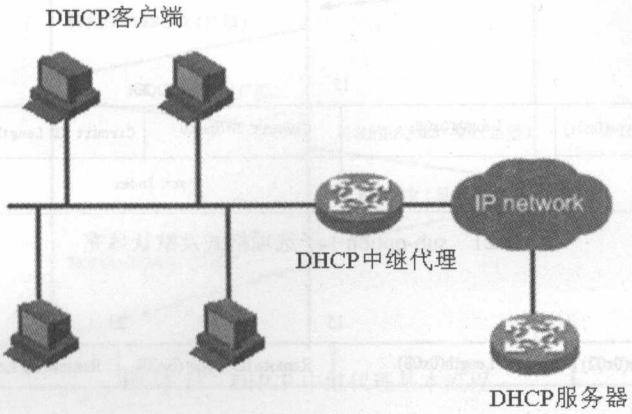


图 11-20　DHCP 中继代理的典型应用示例



其实这里的 DHCP 中继代理就相当于网关，DHCP 广播报文都是通过这个网关来进行转发的。DHCP 中继代理服务的典型应用环境就是一个 DHCP 服务器为所连接的交换机上的多个 VLAN 分配 IP 地址。这时因为 DHCP 服务器只能隶属于一个 VLAN，所以要为多个 VLAN 同时分配 IP 地址的话，就必须在这些 VLAN 与 DHCP 服务器之间有一个 DHCP 中继代理，那就是需要在三层交换机上开启 DHCP 中继代理服务，让三层交换机担当 DHCP 中继代理角色。然后在 DHCP 服务器上启用 82 选项（用于识别 DHCP 中继代理），并在每个地址池中配置一个指向每个与 DHCP 服务器不在同一子网的默认网关地址（各 VLAN 接口的 IP 地址）即可。

1. DHCP 中继代理简介

Option 82 是 DHCP 报文中的中继代理信息选项（Relay Agent Information Option），该选项记录了 DHCP 客户端的位置信息。管理员可以利用该选项定位 DHCP 客户端，实现对客户端的安全和计费等控制。支持 Option 82 选项的 DHCP 服务器还可以根据该选项的信息制订 IP 地址和其他参数的分配策略，提供更加灵活的地址分配方式。

在整个 Option 82 选项中最多又可以包含 255 个子选项，但至少要定义一个子选项。目前设备主要只支持两个子选项：sub-option 1（Circuit ID，电路 ID 子选项）和 sub-option 2（Remote ID，远程 ID 子选项）。但由于 RFC 3046 对于 Option 82 的内容没有统一的规定，不同厂商通常根据需要进行填充。目前，设备作为 DHCP 中继设备都支持 Option 82 子选项的扩展填充格式。

sub-option 1 和 sub-option 2 两个子选项格式分别如图 11-21 和图 11-22 所示，固定字段（相当

于子选项头）括号中的内容为该字段的固定取值。sub-option 1 子选项中的内容是接收到 DHCP 客户端请求报文的端口所属 VLAN 的编号（对应 VLAN ID 字段，占 2 个字节）和端口索引（端口索引的取值为端口物理编号减 1，对应 Port Index 字段，占 2 个字节）。sub-option 2 子选项的内容是接收到 DHCP 客户端请求报文的 DHCP 中继设备的 MAC 地址（对应 MAC Address 字段，占 6 个字节）。

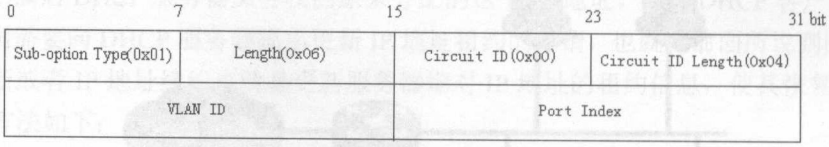


图 11-21　sub-option 1 子选项格式及默认填充

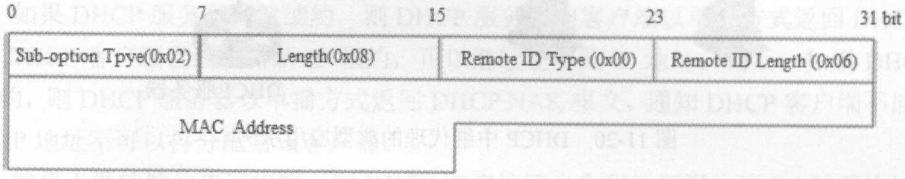


图 11-22　sub-option 2 子选项格式及默认填充

2. 通过 DHCP 中继代理动态分配 IP 地址原理

DHCP 客户端通过 DHCP 中继代理服务从 DHCP 服务器动态获取 IP 地址的过程与 11.3.4 节中介绍的不通过 DHCP 中继代理而直接从 DHCP 服务器动态获取 IP 地址的过程基本相同，都要经历发现、提供、选择和确认 4 个阶段。所用到的报文也对应是 DHCP DISCOVER、DHCP OFFER、DHCP REQUEST、DHCP ACK，只是在这里 DHCP 中继代理需要起到一个中介代理的作用，负责转发 DHCP 客户端与 DHCP 服务器之间交互的这些报文，基本流程如图 11-23 所示。

【说明】在此流程中，DHCP 客户端发送的请求报文有两种：DHCP DISCOVER 报文和 DHCP REQUEST 报文。由于不同厂商生产的 DHCP 服务器设备对请求报文的处理机制不同，有些设备处理 DHCP DISCOVER 报文中的 Option 82 信息，而有些设备处理 DHCP REQUEST 报文中的 Option 82 信息，因此 DHCP 中继设备将在这两种报文中都添加 Option 82 选项。

这里将只介绍 DHCP 中继代理设备支持 Option 82 时的工作机制，具体如下：

（1）DHCP 客户端以广播方式（因为不知道中继代理设备的 IP 地址）向本网段发送 DHCP DISCOVER 或 DHCP REQUEST 请求报文，此时只有网络中的 DHCP 中继代理设备会接收该 DHCP 请求报文（假设本网段中没有 DHCP 服务器）。

（2）DHCP 中继代理设备在收到 DHCP 客户端发来的 DHCP DISCOVER 或 DHCP REQUEST 请求报文后，将检查报文中是否已有 Option 82 选项，主要是上面介绍的 sub-option 1 子选项和 sub-option 2 子选项。

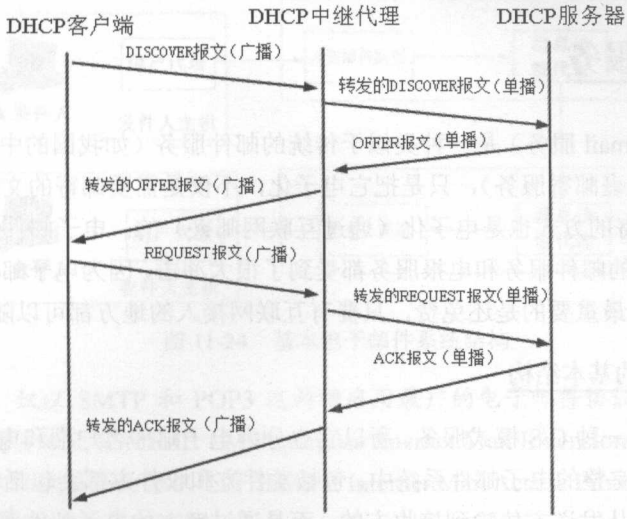


图 11-23　DHCP 中继代理基本流程

如果请求报文中已有 Option 82，DHCP 中继代理设备会按照配置的策略对该报文进行处理（丢弃，或者用中继设备本身的 Option 82 选项替代报文中原有的 Option 82 选项，或者保持报文中原有的 Option 82 选项），同时根据 Option 82 选项的 sub-option 1 子选项中的 VLAN ID 和 Port Index 字段配置找到为对应网段所分配的 DHCP 服务器地址，并将请求报文中的 giaddr 字段填充为 DHCP 中继代理设备的 IP 地址；然后将请求报文以单播方式（因为在中继代理设备中已配置好了对应的 DHCP 服务器地址）转发给指定的 DHCP 服务器。

如果请求报文中没有 Option 82 选项，则 DHCP 中继设备将 Option 82 选项添加到报文中后，根据 Option 82 选项的 sub-option 1 子选项中的 VLAN ID 和 Port Index 字段配置找到为对应网段所分配的 DHCP 服务器地址，并将报文中的 giaddr 字段填充为 DHCP 中继代理设备的 IP 地址；然后同样根据中继代理设备中为对应网段所配置的 DHCP 服务器地址以单播方式将请求报文转发给 DHCP 服务器。

（3）DHCP 服务器在收到由 DHCP 中继代理设备转发的 DHCP DISCOVER 或 DHCP REQUEST 请求报文后，根据转发的请求报文中的 Giaddr 字段值所对应的中继代理设备 IP 地址，以及在 Option 82 选项中的 sub-option 2 子选项中的中继代理 MAC 地址以单播方式向 DHCP 中继代理返回对应的 DHCP OFFER 或 DHCP ACK 应答报文。

（4）DHCP 中继设备在收到 DHCP 服务器的应答报文后，将剥离报文中的 Option 82 信息，然后以广播方式将带有 DHCP 配置信息的对应应答报文转发给 DHCP 客户端，完成对客户端的动态配置。

在以上过程中，4 种 DHCP 报文中除 Option 82 选项和 Giaddr 字段外，其他各字段值均与 11.3.4 节介绍的非中继方式动态分配 IP 地址过程中的 4 种报文所对应的字段值是一样的。

11.5 电子邮件服务

电子邮件服务（E-mail 服务）是一种类似于传统的邮件服务（如我国的中国邮政集团、各家快递公司提供的信件和包裹邮寄服务），只是把它电子化，不仅是需要邮寄的文件要电子化（也就是计算机文件），而且邮寄的方式也是电子化（通过互联网邮寄）的。电子邮件服务自 20 世纪 90 年代得到普及以后，传统的邮件服务和电报服务都受到了很大冲击，因为电子邮件服务比传统邮件服务更快，可靠性更高，最重要的是还免费，只要有互联网接入的地方都可以随时收发电子邮件。

11.5.1 电子邮件系统的基本结构

电子邮件服务也是一种 C/S 模式服务，所以它也分为电子邮件客户端和电子邮件服务器端。但需要注意的是，其实在完整的电子邮件系统中，包括发件方和收件方都会包括一套客户端和服务器端。电子邮件不是直接从发送方传输到接收方的，而是通过双方的电子邮件服务器转发、保存。这时，各自的电子邮件服务器就相当于一个存储电子邮件的仓库和用于非即时电子邮件转发的中转站。这样做有两方面的好处：

- 用户不必时时在线等待那些根本无法预料什么人、什么时候会发来的电子邮件，这个工作直接由自己选择的电子邮件服务器来完成，因为电子邮件服务器，特别是 ISP 提供的电子邮件服务器是 24 小时不关机的。电子邮件服务器可以 24 小时随时接收并保存用户的电子邮件，用户想什么时候收邮件就什么时候收邮件。
- 用户自己不必识别互联网上各种各样的电子邮件域名，这个工作也全部交给由 ISP 提供的电子邮件服务器完成。这样可大大降低邮件用户主机的配置难度，只需安装一个电子邮件客户端软件，然后在客户端中配置好自己购买的 ISP 电子邮箱账户以及对应的电子邮件服务器地址即可。

整个电子邮件系统包括“用户代理”（User Agent）和“消息传输代理”（Message Transfer Agent）两大部分。“用户代理”负责电子邮件用户书写、发送、接收和阅读电子邮件，是安装在用户端计算机上的各种电子邮件客户端软件，是用户与电子邮件系统间交互的本地程序。“消息传输代理”负责电子邮件在发件方和收件方的客户端和服务器端之间，以及在发件人服务器端与收件人服务器端之间的传输，是安装在 ISP 或者本地网络中的各种电子邮件服务器软件。“消息传输代理”也称“守护进程”（daemon），是在服务器程序后台运行的应用服务进程。基本电子邮件系统结构如图 11-24 所示。

与其他计算机网络应用服务一样，电子邮件服务也是由一些具体的应用层协议来实现的。目前提供电子邮件发送服务的协议主要是 SMTP（Simple Mail Transfer Protocol，简单邮件传输协议），而提供邮件接收服务的协议主要是 POP3（Post Office Protocol 3，邮局协议 3）。它们都是工作于 C/S 模式的，都有对应的客户端和服务器端两部分。在“用户代理”部分会同时提供 SMTP 客户端和 POP3 客户端服务，而在“消息传输代理”部分则除了包括 SMTP 服务器和 POP3 服务器外，还要包括 SMTP 客户端，因为在发送和接收邮件两方的邮件服务器之间还需要进行电子邮件传输。

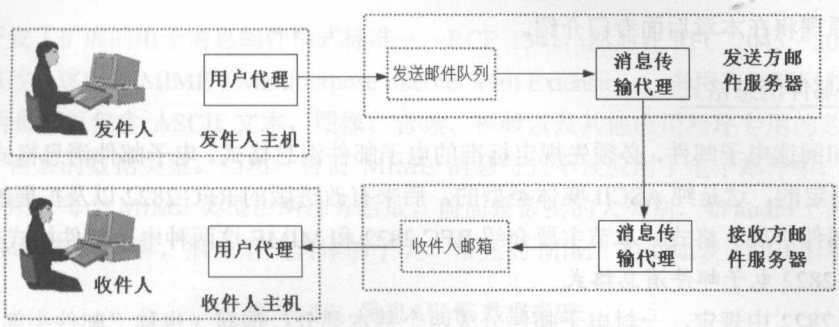


图 11-24　基本电子邮件系统结构

【说明】在此，仅以 SMTP 和 POP3 这两种应用最广的电子邮件协议进行介绍。除此之外，提供邮件发送服务的协议还有 MIME（Multipurpose Internet Mail Extensions，多用途互联网邮件扩展）协议，提供邮件接收服务的协议还有 IMAP（Internet Mail Access Protocol，互联网邮件访问协议）。具体也将在本章后面有简单介绍。

图 11-25 显示了一个完整的 SMTP/POP3 电子邮件传输系统中，发件方和收件方各自电子邮件系统的基本组成。在发件人和收件人计算机中会安装各种用于电子邮件收发的电子邮件客户端软件，也就是“用户代理”，如 Outlook、Foxmail 等。在这类软件中会全面包括 SMTP 客户端和 POP3 客户端程序，因为一般用户不可能通过此程序只发送或者只接收电子邮件。

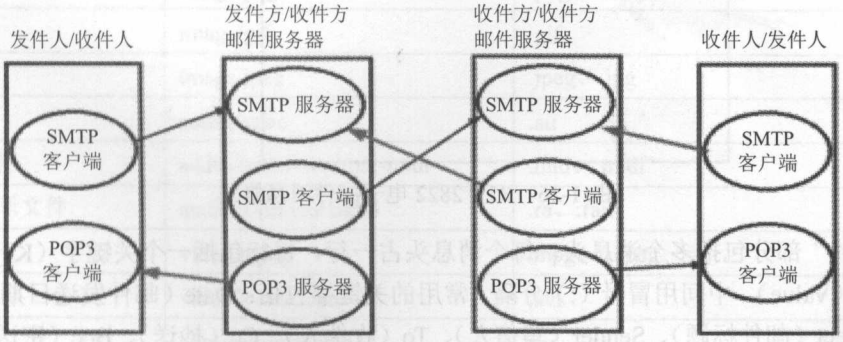


图 11-25　SMTP/POP3 电子邮件传输系统

在发送方和接收方邮件服务器中会安装各种电子邮件服务器软件，也就是“消息传输代理”部分，如 Exchange Server、SendMial、WinMail、CCMail 等。在邮件服务器软件中会同时提供 SMTP 服务器、POP3 服务器和 SMTP 客户端，因为一个电子邮件服务器既要把由用户主动发送上来的电子邮件转发到接收方的电子邮件服务器中（需要同时利用 SMTP 客户端和 SMTP 服务器），又要允许用户从电子邮件服务器上取回自己的电子邮件（利用的是 POP3 客户端）。

SMTP 和 POP3 服务都是使用 TCP 传输层协议进行邮件传输的，各自使用的端口分别为 TCP 25 和 TCP 110。在图 11-25 所示的电子邮件系统中电子邮件的发送和接收都是通过 TCP 协议进行的，

具体的工作原理将在本章后面专门介绍。

11.5.2 电子邮件消息格式

要书写和阅读电子邮件，必须先规定标准的电子邮件消息格式。电子邮件消息格式最开始是在RFC 822 中规定的，这是纯 ASCII 媒体类型的，后来有改进版的 RFC 2822 以及扩展的 MIME（多用户互联网邮件扩展）格式。本节主要介绍 RFC 2822 和 MIME 这两种电子邮件格式标准。

1. RFC 2822 电子邮件消息格式

在 RFC 2822 中规定，一封电子邮件分成两个基本部分：信封（也称“邮件头部”）和邮件正文，如图 11-26 所示，其实它就是一个表单。

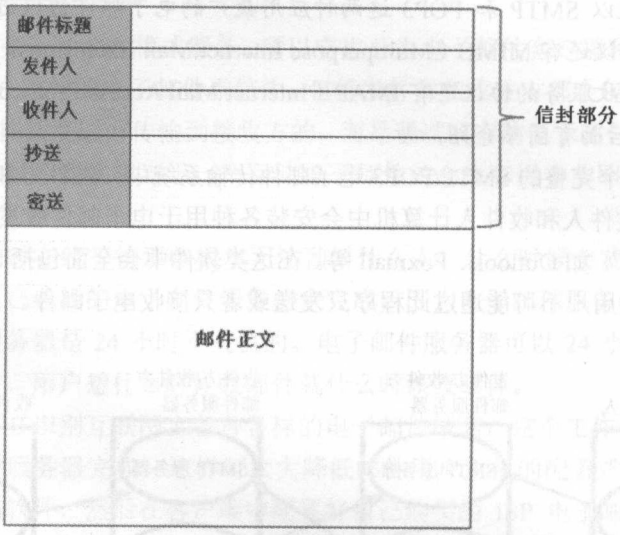


图 11-26 RFC 2822 电子邮件消息格式

在“信封”部分包括多个消息头，每个消息头占一行，每行包括一个关键字（Keyword）和一个对应的值（Value），中间用冒号（:）分隔。常用的关键字包括：Date（邮件发送日期）、From（发件人）、Subject（邮件标题）、Sender（写信人）、To（收件人）、Cc（抄送）、Bcc（密送）、Reply-To（回信地址）等。Subject 关键字后，每次回复都在前面加上一个 Re，从中可以看出这是第几次回复；Reply-To 关键字后面所跟的邮箱地址可以与 From 关键字后面跟的发信人邮箱地址相同，也可以不同，因为有的用户是有不同的发送邮件地址和接收邮件地址的；在“信封”部分中每个关键字和所对应的值独占一行，在结尾都有一个回车和换行符（GRLF）。

“邮件正文”部分是具体的邮件内容，但 RFC 2822 仅支持 ASCII 文本字符。

2. MIME 电子邮件格式

前面介绍的 RFC 2822 只能传输 ASCII 码格式的数据，一些非英语字符消息，像二进制文件、图像、声音等非文字消息都不能在电子邮件中传输。这显然不能满足当前多媒体时代的电子邮件通信

需求，于是开发了扩展的电子消息邮件格式标准——RCF 1341，然后在 RFC 2045～2049 多个 RFC 文档中做了修改，这就是 MIME（Multipurpose Internet Mail Extensions，多用户互联网邮件扩展）。

MIME 消息可以包含 ASCII 文本、图像、音频、视频以及其他应用程序专用的数据，大大扩展了电子邮件消息的数据类型。当然，目前 MIME 消息格式不仅应用于电子邮件中，还在各种浏览器中普遍应用。每个 MIME 类型由两部分组成，前面是数据的大类别，如 audio（音频）、image（图像）、video（视频）等，后面定义具体的子类。常见的 MIME 类型如表 11-10 所示。

表 11-10 常见 MIME 数据类型

MIEM 类别	MIME 大类/MIME 子类	扩展名
XML 文档	text/xml	.xml
XHTML 文档	application/xhtml+xml	.xhtml
普通文本	text/plain	.txt
超文本标记语言文本	text/html	.html
RTF 文本	application/rtf	.rtf
PDF 文档	application/pdf	.pdf
Word 文档	application/msword	.doc
Excel 文档	application/msexcel	.xls
PNG 图像	image/png	.png
GIF 图形	image/gif	.gif
JPEG 图形	image/jpeg	.jpeg、.jpg
au 声音文件	audio/basic	.au
MIDI 音乐文件	audio/midi，audio/x-midi	.mid、.midi
RealAudio 音乐文件	audio/x-pn-realaudio	.ra、.ram
MPEG 文件	video/mpeg	.mpg、.mpeg
AVI 文件	video/x-msvideo	.avi
GZIP 文件	application/x-gzip	.gz
TAR 文件	application/x-tar	.tar

在电子邮件格式方面，MIME 也继承了 RFC 2822 的格式，分为“信封”和“邮件正文”两部分。在“信封”部分除了仍支持 RFC 2822 中的消息头外，还新增了如表 11-11 所示的消息头。

表 11-11 MIME 新增的消息头

新增的 MIME 消息头	说明
MIME-Version	标识使用的 MIME 协议版本，目前都是 1.0 版本
Content-Type	指示消息中的数据类型，以便数据能被适当的处理，参见表 11-10

续表

新增的 MIME 消息头	说明
Content-Transfer-Encoding	指示消息中数据所采用的编码格式。可以使用 7bit(7 位元的 ASCII 编码方式)、8bit（8 位元的 ASCII 编码方式)、binary（二进制编码方式)、quoted-printable 和 base64 中的一种编码方式
Content-ID	用于为 multipart/related 组合消息中的内嵌资源指定一个唯一标识号，在 HTML 格式的正文中可以使用这个唯一标识号来引用该内嵌资源
Content-Description	指示邮件阅读程序处理数据内容的方式，有 inline 和 attachment 两种标准方式，inline 表示直接处理，而 attachment 表示当作附件处理。如果将 Content-Disposition 设置为 attachment，在其后还可以指定 filename 属性
Content-Location	用于为内嵌资源设置一个 URI 地址，这个 URI 地址可以是绝对的或相对的。当使用 Content- Location 头字段为一个内嵌资源指定一个 URI 地址后，在 HTML 格式的正文中也可以使用这个 URI 来引用该内嵌资源
Content-Base	用于为内嵌资源设置一个基准路径，只有这样，Content-Location 头字段中设置的 URI 才可以采用相对地址

11.5.3 SMTP 请求命令和应答消息

SMTP（对应 RFC 2821）是目前应用最广泛的提供电子邮件发送服务的应用层协议，使用 TCP 端口 25，但它是仅可以进行 ASCII 字符传输的协议（注意，这里并不是说邮件内容仅可以是 ASCII 字符，而仅是指 SMTP 通信时是采用 ASCII 消息格式)。

前面说了，SMTP 也是工作在 C/S 模式的，所以它也有对应的客户端和服务器端。但需要注意的是，虽然 SMTP 服务器仅是由各种电子邮件服务器软件提供，但 SMTP 客户端会同时在电子邮件客户端软件和电子邮件服务器软件中提供（参见图 11-24)，因为整个电子邮件发送过程中除了发件人需要利用 SMTP 客户端向自己的邮件服务器发送邮件外，在发件人的电子邮件服务器把邮件传送到收件人的电子邮件服务器的过程中也要利用 SMTP 客户端。

在发送邮件时，源机器的 SMTP 客户端程序要与目的机器的 SMTP 服务器程序 TCP 25 号端口建立 TCP 传输连接。SMTP 客户端是以一系列 4 个字母的 ASCII 格式请求命令向 SMTP 服务器发送请求消息的，而 SMTP 服务器是以一些非常简单的三位数字格式的 ASCII 消息代码进行应答的，不同的消息代码代表不同的含义。

SMTP 邮件发送过程是在基于 25 号 TCP 端口传输连接建立之后进行的，整个邮件传输过程中 SMTP 客户端和 SMTP 服务器之间的会话也可以分为：连接建立、邮件传输和连接释放这 3 个阶段。在这 3 个阶段中，每个阶段 SMTP 客户端使用的请求命令如表 11-12 所示（目前可用的是 11 条命令)，每个命令后面都跟一个 CRLF（回车和换行标识符)，SMTP 服务器的应答消息如表 11-13 所示，SMTP 服务器针对 SMTP 客户端的各个请求命令可以做出的应答消息如表 11-14 所示。

表 11-12 SMTP 客户端请求命令

SMTP 会话阶段	请求命令	说明
连接建立阶段	HELO	后面跟 SMTP 客户端域名，发送人向服务器标识用户身份，请求建立 SMTP 应用会话。命令格式为：HELO （这里要空一格）<域名><CRLF>，如 HELO lycb.com
	EHLO	与 HELO 功能一样，但支持 SMTP 服务扩展的客户应该以 EHLO 命令开始 SMTP 会话，而不是通常的 HELO 命令。命令格式为：EHLO （这里要空一格）<域名><CRLF>，如 EHLO lycb.com
邮件传输阶段	MAIL FROM	后面跟发件人的邮箱地址，向对方服务器标识发件人此次所使用的邮箱地址。命令格式为：MAIL FROM:<reverse-path><CRLF>，reverse-path 包括发件人的邮箱地址，如 MAIL FROM:<winda@lycb.com>
	RCPT TO	后面跟接收此邮件的邮箱地址，向对方服务器标识单个收件人邮箱地址。常在 MAIL From 命令后面可有多个 RCPT To 命令。命令格式为：RCPT TO：<forward-path><CRLF>，forward-path 为收件人邮箱地址，如 grfw@vd.bar.org
	DATA	后面无参数，表示进行邮件传输初始化，准备开始正式的邮件传输过程，在内容的最后以 CRLF 结尾，表示邮件传输结束。命令格式为：DATA<CRLF>，如 DATA
	VRFY	后面跟要确认的用户/邮箱，请求服务器验证指定的用户账户或邮箱是否存在，考虑到用户邮箱账户安全性因素，服务器一般禁止此操作。命令格式为：VRFY（这里要空一格）<校验的字符串><CRLF>，如 VRFY winda@lycb.com
	EXPN	后面跟要确认的邮箱列表，请求服务器验证指定的邮箱列表是否存在，同样考虑到用户邮箱账户安全性因素，服务器一般禁止此操作。命令格式为：EXPN（这里要空一格）<校验的字符串><CRLF>，如 EXPN Example-People
	HELP	后面跟要请求帮助的命令，请求服务器提供帮助信息。命令格式为：HELP（这里要空一格）<要查询帮助信息的命令><CRLF>，如 HELP NOOP（就是想向服务器查询 NOOP 命令的帮助信息）
	NOOP	后面无参数，无操作，要求服务器应答 OK，用于测试。命令格式为：NOOP<CRLF>，如 NOOP
	RSET	后面无参数，重置 SMTP 会话，当前传输被取消。命令格式为：RSET<CRLF>，如 RSET
连接释放阶段	QUIT	后面无参数，终止邮件会话，要求服务器应答 OK。命令格式为：QUIT<CRLF>，如 QUIT

【说明】在 RFC 822 中，SMTP 还有几条已废弃的命令，其中 SEND、SOML 和 SAML 现在都直接用 MAIL FROM 命令替代，TURN 被 QUIT 命令所取代。

表 11-13　SMTP 服务器应答消息

消息代码	说明
200	非标准的成功应答，参见 RFC 876
211	显示服务器系统状态或系统帮助信息
214	显示帮助消息
220	显示的域 SMTP 服务已准备就绪
221	显示的域 SMTP 服务关闭了传输信道
250	操作正确完成，OK
251	用户非本地，将转发到显示的域中
354	开始邮件输入，以<CRLF>.<CRLF>结束，CRLF 为回车和换行标识符
421	指定域的 SMTP 服务不可用，关闭传输信道
450	请求的邮件操作未完成，邮箱不可用
451	服务器本地处理进程出错，放弃请求的邮件操作
452	服务器系统存储空间不足，请求的操作未执行
500	语法错误，命令不可识别
501	命令参数或格式语法错误
502	命令不可执行
503	错误的命令序列
504	命令参数未执行
521	显示的域不接受邮件
530	拒绝访问
550	邮箱不可用，请求的邮件操作不执行
551	非服务器本地用户，请尝试所显示的域
552	超出了对应用户的邮件存储空间分配，放弃请求的邮件操作
553	用户邮箱名不允许，不执行请求的邮件操作
554	邮件传输失败

表 11-14　各 SMTP 请求命令可能返回的应答消息

SMTP 客户端请求命令	可能返回的消息代码	说明
HELO	250	请求的邮件操作正确完成，OK
	500	语法错误，命令不可识别
	501	命令参数语法错误

续表

SMTP 客户端请求命令	可能返回的消息代码	说明
HELO	504	命令参数未执行
	521	显示的域没有接受邮件
	421	显示域的 SMTP 服务不可用，关闭传输信道
EHLO	250	请求的邮件操作正确完成，OK
	550	不执行
	500	语法错误，命令不可识别
	501	命令参数语法错误
	504	命令参数未执行
	421	显示域的 SMTP 服务不可用，关闭传输信道
MAIL	250	请求的邮件操作正确完成，OK
	552	超出用户邮件存储空间限制，请求的邮件操作放弃
	451	服务器本地进程错误，请求的邮件操作放弃
	452	服务器系统存储空间不足，请求的邮件操作未发生
	500	语法错误，命令不可识别
	501	命令参数语法错误
	421	显示域的 SMTP 服务不可用，关闭传输信道
RCPT	250	请求的邮件操作正确完成，OK
	251	非服务器上的本地用户，将转发到显示的邮箱地址
	550	邮箱不可用，请求的邮件操作未发生
	551	非服务器上的本地用户，将尝试显示的邮箱地址
	552	超出用户存储空间分配限制，请求的邮件操作放弃
	553	用户邮箱名不允许，请求的邮件操作未发生
	450	用户邮箱不可用，请求的邮件操作未发生
	451	本地进程错误，请求的邮件操作放弃
	452	服务器系统存储空间不足，请求的邮件操作未发生
	500	语法错误，命令不可识别
	501	命令参数语法错误
	503	错误的命令序列
	521	显示的邮件域不接受邮件
	421	显示域的 SMTP 服务不可用，关闭传输信道

续表

SMTP 客户端请求命令	可能返回的消息代码	说明
DATA	354	开始邮件传输
	451	本地进程错误，请求的邮件操作放弃
	554	传输失败
	500	语法错误，命令不可识别
	501	命令参数语法错误
	503	错误的命令序列
	421	显示域的 SMTP 服务不可用，关闭邮件接收传输信道
	250	请求的邮件操作正确完成，OK
	552	超出用户存储空间分配限制，请求的邮件操作放弃
	452	服务器系统存储空间不足，请求的邮件操作未发生
RSET	200	非标准成功应答
	250	请求的邮件操作正确完成，OK
	500	语法错误，命令不可识别
	501	命令参数语法错误
	504	命令参数未执行
	421	显示的域 SMTP 服务不可用，关闭邮件接收传输信道
VRFY	250	请求的邮件操作正确完成，OK
	251	非服务器上的本地用户，将转发到显示的邮箱地址
	550	邮箱不可用，请求的邮件操作未发生
	551	非服务器上的本地用户，将尝试显示的邮箱地址
	553	用户邮箱名不允许，请求的邮件操作未发生
	500	语法错误，命令不可识别
	501	命令参数语法错误
	502	命令未执行
	504	命令参数未执行
	421	显示域的 SMTP 服务不可用，关闭邮件接收传输信道
EXPN	250	请求的邮件操作正确完成，OK
	550	邮箱不可用，请求的邮件操作未发生
	500	语法错误，命令不可识别
	501	命令参数语法错误
	502	命令未执行

续表

SMTP 客户端请求命令	可能返回的消息代码	说明
EXPN	504	命令参数未执行
	421	显示域的 SMTP 服务不可用，关闭邮件接收传输信道
HELP	211	系统状态或系统帮助应答
	214	帮助消息
	500	语法错误，命令不可识别
	501	命令参数语法错误
	502	命令未执行
	504	命令参数未执行
	421	显示域的 SMTP 服务不可用，关闭邮件接收传输信道
NOOP	200	非标准成功应答
	250	请求的邮件操作正确完成，OK
	500	语法错误，命令不可识别
	421	显示域的 SMTP 服务不可用，关闭邮件接收传输信道
QUIT	221	显示域的 SMTP 服务不可用，关闭传输信道
	500	语法错误，命令不可识别

11.5.4 SMTP 服务工作原理

SMTP 服务工作在两种情况下：一是从发送方把电子邮件传输到发送方邮件服务器上；二是从发送方邮件服务器传输到接收方邮件服务器上。它们都是通过两端直接建立的 TCP 传输连接进行邮件传输的，接收邮件的 SMTP 服务器端使用 TCP 25 号端口。

1. SMTP 邮件传输的 3 个阶段

SMTP 客户端使用“推”（Push）的方式发送邮件，每个邮件传输的 SMTP 服务应用过程均包括：连接建立（这是指 SMTP 应用会话连接建立阶段，不是指 TCP 传输连接建立阶段）、邮件传输和连接释放 3 个阶段（注意这 3 个阶段都是在 TCP 传输连接建立后进行的）。有关这 3 个阶段所用的请求命令的应答消息参见上节中的介绍，下面是这 3 个阶段的基本流程。其中的“发送方”既可以是发件人主机，也可以是发送方电子邮件服务器，“接收方”既可以是发送方电子邮件服务器，也可以是接收方电子邮件服务器，具体要视所进行的邮件传输过程是从发件人主机发送到发送方电子邮件服务器，还是从发送方电子邮件服务器到接收方电子邮件服务器。

（1）连接建立阶段。

在 SMTP 应用会话连接建立阶段，当发送方有邮件发送时，首先安装在其主机中的 SMTP 客户端程序会与所配置的邮件服务器建立好基于 TCP 25 号端口的传输连接。以后所有的邮件传输过程都是在已建立好的 TCP 连接基础上进行的。

TCP 传输连接建立好后，发送方调用 SMTP 客户端服务发送 HELO 命令，后面跟发送方的 DNS 域名，向接收方标识自己的身份。如果能通过接收方的身份验证，接收方会返回一条类似于“250 OK”的应答消息，表示接收方已接受会话连接建立请求，建立好了会话连接；如果在发送连接请求的 HELO 命令中出现了差错或者接收方当前 SMTP 服务不可用，则会显示对应的错误代码，具体参见表 11-14。

（2）邮件传输阶段。

发件方在发送完 HELO 命令并且收到接收方返回的 250 消息代码后，即表明接收方已准备就绪，进入到了邮件传输阶段，发送方可以立即通过 SMTP 客户端发送邮件了。在邮件传输阶段，SMTP 客户端可以使用多个命令，如用 MAIL FROM 命令指示发件人的邮箱地址，用 RCPT TO 命令指示收件人的邮箱地址，并且可以发送多条 RCPT TO 命令，每条命令包括一个收件人邮箱地址。

当以上两条命令发送后，发送方如果都收到了接收方返回的类似于“250 OK”的应答消息，即表示接收方已正确执行了该命令，发送方 STMP 客户端就可以通过 DATA 命令进行正式的邮件内容传输了。在邮件内容全部传输完后，再传输一条<CRLF>.<CRLF>消息，以标识邮件内容的结束。接收方在收到最后一条消息后会返回类似于“250 OK”的应答消息代码。

如果有另外一封邮件要传输，它会继续重复上面的邮件传输过程，但无需重复连接建立阶段。

（3）连接释放阶段。

当没有邮件传输时，发送方 SMTP 客户端要发送一条 QUIT 命令结束本次 SMTP 应用会话进程。此时，正常情况下接收方会返回一条代码为 221 的应答消息，表示服务器已接受关闭连接请求，释放本次 SMTP 应用会话连接。

2. SMTP 邮件传输流程示例

为了理解以上邮件传输流程，下面仅以发送方邮件服务器把邮件传送到接收方邮件服务器的过程为例介绍 SMTP 邮件传输过程中的 3 个阶段。现假设发送方邮件服务器的域名为 lycb.com，发件人的邮箱地址为 winda@lycb.com，发送方邮件服务器的域名为 lycb.com，接收方邮件服务器的域名为 grfw.com，基本过程如图 11-27 所示（仅考虑基本流程和正常情况下的应答代码，不考虑各种错误）。

下面是一个典型的 SMTP 邮件传输代码脚本示例，其中 C 表示 SMTP 客户端发送的请求命令，S 代表 SMTP 服务器返回的应答消息。其中各请求命令和应答消息代码的含义参见上节中的相关内容。

```
S: 220 foo.com Simple Mail Transfer Service Ready    !---SMTP 服务器监听到 25 号 TCP 端口上有连接请求，建立
!---好相应的 TCP 连接并通知 SMTP 客户端，说它已做好了准备，可以正式进行邮件传输了
C: <open connection>   !---SMTP 客户端打开已建立的 TCP 连接
C: HELO bar.com
S: 250-foo.com greets bar.com
S: 250-8BITMIME
S: 250-SIZE
S: 250-DSN
S: 250 HELP
C: MAIL FROM:<Smith@bar.com>
S: 250 OK
```

```
C: RCPT TO:<Jones@foo.com>
S: 250 OK
C: RCPT TO:<Green@foo.com>
S: 550 No such user here
C: RCPT TO:<Brown@foo.com>
S: 250 OK
C: DATA
S: 354 Start mail input; end with <CRLF>.<CRLF>
C: Blah blah blah...
C: ...etc. etc. etc.
C: .
S: 250 OK
C: QUIT
S: 221 foo.com Service closing transmission channel
```

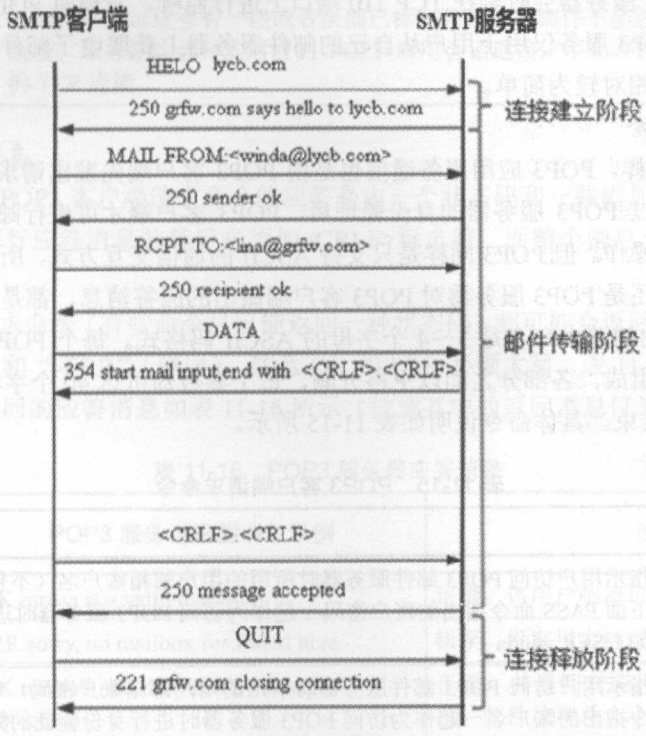


图 11-27 SMTP 邮件传输 3 个阶段的基本请求命令和应答流程

11.5.5 POP3 请求命令及应答消息

上面介绍的 SMTP 服务采用“推”的方式可以把邮件发送到发送方自己的邮件服务器，然后由发送方的电子邮件服务器转发到接收方的电子邮件服务器上。但是接收方的电子邮件服务器不会通过 SMTP 服务再把邮件推送到收件人的主机上，因为收件人并不知道什么时候有人发邮件给他，

也不可能 24 小时在线，不可能随时接收。我们希望的是像发件人发送邮件那样，在需要的时候随时进行。这时就得靠 POP3 这样具有“拉”（Pull）功能的邮件服务了。

POP3（对应 RFC 1939）是 POP（对应 RFC 918）协议的第 3 个版本，解决了 POP 服务器在用户阅读了某封邮件后即删除该邮件的不足，POP3 服务器允许用户把邮件存储在邮件服务器上。POP3 允许用户从服务器上把邮件存储到本地主机（即自己的计算机）上，同时根据客户端的需要删除或保存在邮件服务器上的邮件。如果我们把 SMTP 服务器称为“发送邮件服务器”，是用来发送邮件的，则 POP3 服务器就是“接收邮件服务器”，是用来接收邮件的。

POP3 也是工作在 C/S 模式，POP3 客户端位于安装了各种电子邮件客户端软件的用户主机上，而 POP3 服务器则位于安装了各种电子邮件服务器软件的服务器主机上，参见图 11-26 所示的电子邮件传输系统。POP3 客户端和 POP3 服务器之间的通信是在 TCP 110 号端口所建立的 TCP 传输连接基础之上的。POP3 服务器会时刻在 TCP 110 端口上进行监听，发现有 POP3 客户连接请求即做出相应的应答。但 POP3 服务仅用于用户从自己的邮件服务器上获取电子邮件，不像 SMTP 服务那样涉及到两个过程，相对较为简单。

1. POP3 请求命令

与 SMTP 服务一样，POP3 应用服务通信也是由 POP3 客户端先发出请求命令，POP3 服务器返回应答消息。在通过 POP3 服务器的身份验证后，POP3 客户端才可进行邮件接收或者从 POP3 服务器上删除邮件等操作。但 POP3 同样是只支持 ASCII 的通信交互方式，所以无论是 POP3 客户端发出的请求命令，还是 POP3 服务器对 POP3 客户端做出的应答消息，都是 ASCII 码格式。

POP3 客户端发出的请求命令是 3～4 个字母的 ASCII 码格式。每个 POP3 请求命令由一个命令关键字和一些参数组成，各部分之间以空格分隔，每个参数却可达 40 个字符长，所有命令均以一个<CRLF>标志符结束。具体命令说明如表 11-15 所示。

表 11-15　POP3 客户端请求命令

命令	说明
USER name	指示用户访问 POP3 邮件服务器时所用的用户邮箱账户名（不包括域名部分），它与下面 PASS 命令指出的账户密码一起作为访问 POP3 服务器时进行身份验证的凭据，如 USER winda
PASS string	指示用户访问 POP3 邮件服务器时所用的用户邮箱账户密码，它与上面的 USER 命令指出的账户名一起作为访问 POP3 服务器时进行身份验证的凭据
APOP name digest	指示用户访问 POP3 服务器所使用的加密密钥（避免以明文方式在网上传输密码），name 参数是用户邮箱账户名，digest 为通过 MD5 协议加密后的密钥
STAT	查询用户自己的邮箱统计信息。最开始显示的是 OK，然后空一格后再显示邮件的总数，再空一格后显示所有邮件的大小等
LIST [msg]	请求查看指定（选择 msg 参数时）或者全部（不选择 msg 参数时）邮件汇总信息，服务器应答后会显示每个列出的邮件的邮件号和对应的邮件大小
RETR msg	请求接收指定邮件

续表

命令	说明
DELE msg	请求给指定邮件做删除标记。做了删除标记的邮件不会被真正删除，仅在 POP3 客户端发出 QUIT 命令后才会真正删除这些做了删除标记的邮件
NOOP	无操作，用于检测 POP3 客户端在 POP3 服务器端的连接情况，仅用于测试
RSET	恢复所有邮件均为无删除标记，也就是清除所有通过 DELE 命令所做的邮件删除标记
TOP msg lines	请求查看指定邮件（通过 msg 参数指定）中指定行（通过 lines 参数指定）前的内容
UIDL [msg]	请求查看指定（选择 msg 参数时）或邮箱中所有（不选择 msg 参数时）邮件的内容，但不能查看有删除标记的邮件
QUIT	用户邮件接收完成，请求结束本次 POP3 应用会话。如果此时某邮件消息打上了删除标记，则服务器会删除该邮件（不在服务器上保留备份）；如果在删除邮件时发生了错误，则可能会有一些或者全部已标记为删除的邮件不会被删除。无论是否删除成功，服务器均会释放对应的 POP3 应用会话连接，不允许访问了，同时关闭对应的 TCP 连接

2. POP3 应答消息

POP3 服务器对 POP3 客户端请求命令的应答是由一个状态码和一些附加信息组成的，应答消息可以占多行，但每行应答消息的最后必须加<CRLF>标志符，而整个消息要以<CRLF>.<CRLF>结束。

大多数 POP3 请求命令（有些命令只可能返回一种状态码）都可能会返回两种不同的应答状态码：“+OK”（确定）和“-ERR”（错误）。注意 OK 和 ERR 必须大写。表 11-15 中所示的 POP3 客户端请求命令可能返回的应答消息如表 11-16 所示（注意其中的返回消息仅为示例）。

表 11-16 POP3 服务器应答消息

POP3 客户端请求命令	POP3 服务器应答消息示例	说明
USER	+OK mrose is a valid mailbox	正确，该用户邮箱可用
	-ERR sorry, no mailbox for frated here	错误，没有指定的邮箱用户
PASS	+OK mrose's maildrop has 2 messages (320 octets)	正确，并返回对应用户邮箱的基本统计信息，邮件数和总大小
	-ERR maildrop already locked	错误，指定用户邮箱已被锁定
APOP	+OK maildrop has 1 message (369 octets)	正确，并返回对应用户邮箱的基本统计信息，邮件数和总邮件大小
	-ERR permission denied	错误，拒绝访问
STAT	+OK 2 320	确认，返回对应用户邮箱的邮件数和总大小
LIST	+OK 2 messages (320 octets)	正确，返回对应邮件的邮件号和大小，并开始接收邮件

续表

POP3 客户端请求命令	POP3 服务器应答消息示例	说明
LIST	-ERR no such message, only 2 messages in maildrop	错误，没有对应邮件号的邮件
RETR	+OK 120 octets	正确，返回对应邮件大小
	-ERR no such message	错误，没有对应邮件号的邮件
DETE	+OK message 1 deleted	正确，返回指定邮件已被标记为删除
	-ERR message 2 already deleted	错误，指定邮件已被标记为删除
NOOP	+OK	确认，无返回
RSET	+OK maildrop has 2 messages (320 octets)	确认，返回已恢复删除标记的邮件数和总大小
TOP	+OK	正确，返回指定行的邮件内容
	-ERR no such message	错误，没有对应邮件内容
UIDL	+OK 2 QhdPYR:00WBw1Ph7x7	正确，返回对应邮件号的唯一 ID 号
	-ERR no such message, only 2 messages in maildrop	错误，没有对应邮件号的邮件
QUIT	+OK dewey POP3 server signing off	确认，对应服务器正在释放 POP3 应用会话连接

11.5.6 POP3 服务工作原理

理论上来讲，也可以把整个 POP3 邮件接收过程分为“连接建立”“邮件接收”和“连接释放”这 3 个阶段。它们都是在基于与 POP3 服务器 TCP 110 端口建立 TCP 传输连接的基础上进行的，也就是在进行正式的邮件接收前 POP3 客户端必须与 POP3 服务器的 TCP 110 端口建立传输连接（POP3 服务器会时刻监听 TCP 110 端口，一旦有连接请求会做出应答的）。

连接建立阶段其实就是 POP3 客户端向 POP3 服务器发出身份验证请求，给出用户名和密码的阶段，包括表 11-15 所示的 USER、PASS 和 APOP 这几个请求命令。

邮件接收阶段是用户在进行邮件接收时所进行的各种操作，包括表 11-15 中除 USER、PASS、APOP 和 QUIT 这 4 个命令外的其他所有命令。

连接释放阶段是 POP3 客户端在接收完全部邮件或者进行完所有需要的操作后释放本次 POP3 应用会话连接的过程，包括的请求命令就是 QUIT。

图 11-28 所示是一个典型 POP3 邮件接收的基本流程示例，对应的代码脚本如下所示，其中 C 代表 POP3 客户端发送的请求命令，S 代表 POP3 服务器返回的应答消息，其他各请求命令和应答消息代码的含义参见上节中的相关内容：

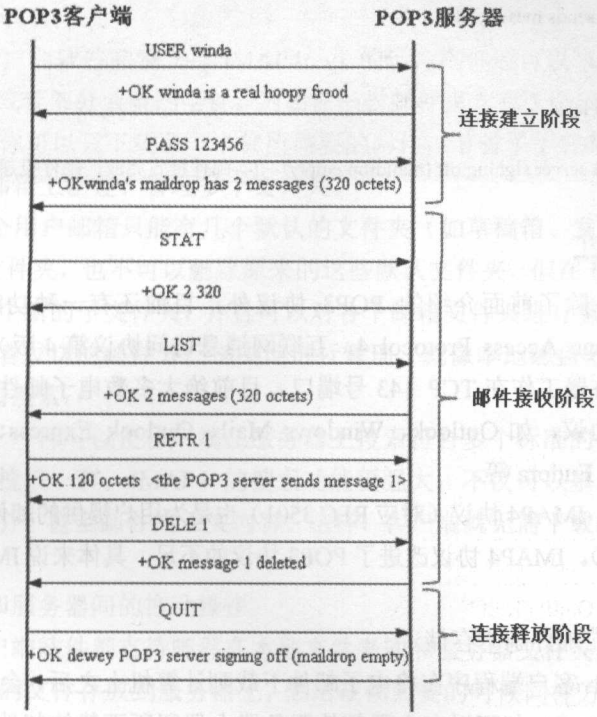


图 11-28 POP3 邮件接收基本流程示例

```
S:    +OK POP3 server ready <1896.697170952@dbc.mtview.ca.us>    !---POP3 服务器监听到 110 号 TCP 端口上有
      !---连接请求，建立好相应的 TCP 连接并通知 POP3 客户端，说它已做好了准备，可以正式进行邮件接收了
C: <open connection>    !---POP3 客户端打开已建立的 TCP 连接
C:    USER winda
S:    +OK winda is a real hoopy frood
C:    PASS 123456
S:    +OK winda's maildrop has 2 messages (320 octets)
C:    STAT
S:    +OK 2 320
C:    LIST
S:    +OK 2 messages (320 octets)
S:    1 120
S:    2 200
S:    .
C:    RETR 1
S:    +OK 120 octets
S:    <the POP3 server sends message 1>
S:    .
C:    DELE 1
S:    +OK message 1 deleted
C:    RETR 2
S:    +OK 200 octets
```

```
S:    <the POP3 server sends message 2>
S:    .
C:    DELE 2
S:    +OK message 2 deleted
C:    QUIT
S:    +OK dewey POP3 server signing off (maildrop empty)   !---邮件接收完成，邮件投递箱为空
```

11.5.7 IMAP4 协议简介

在接收邮件方面，除了前面介绍的 POP3 协议外，目前还有一种功能更强大但也更复杂的 IMAP4（Internet Message Access Protocol 4，互联网消息访问协议第 4 版）。IMAP4 也是采用 C/S 工作模式，IMAP4 服务器工作在 TCP 143 号端口。目前绝大多数电子邮件客户端程序都同时支持 POP3 协议和 IMAP4 协议，如 Outlook、Windows Mail、Outlook Express、Foxmail、Entourage、Mozilla Thunderbird 和 Eudora 等。

与 POP3 协议类似，IMAP4 协议（对应 RFC 3501）也是为用户提供的邮件接收服务的（发送邮件同样是使用 SMTP 协议）。IMAP4 协议改进了 POP3 协议的不足，具体来说 IMAP4 协议具有以下几方面的特性：

（1）支持服务器端邮件副本存储。

默认情况下，POP3 客户端程序在将电子邮件下载到计算机上之后，会从服务器上删除所有已下载的电子邮件。而 IMAP4 却可以在电子邮件服务器上保留所下载的邮件的副本，这样就可以从多台计算机访问到保存在服务器上的同一封电子邮件。

（2）支持离线、在线和断线 3 种访问方式。

POP3 协议仅支持离线访问方式，客户端只会与 POP3 服务器连接一段时间，直到它下载完所有新信息，客户端即断开连接，而 IMAP4 却同时支持离线、在线和断线 3 种访问方式。

在离线方式中，IMAP4 客户软件也是把邮件存储在本地硬盘上，但 IMAP4 提供的摘要浏览功能可以让你通过先阅读完所有的邮件到达时间、主题、发件人、大小等信息后才作出是否下载的决定。也就是说，你不必等所有的邮件都下载完毕后才知道究竟邮件里都有些什么。如果你根据摘要信息就可以决定某些邮件对你毫无用处，你就可以直接在服务器上把这些邮件删除掉，而不必浪费你宝贵的上网时间。在线模式中，IMAP4 客户端可以通过 Web 浏览器来访问 IMAP4 服务器上的邮件，不需要像离线模式那样把邮件下载到本地才能阅读。在断线方式中，IMAP4 客户软件把用户选定的消息和附件复制或缓存到本地磁盘上，并把原始副本留存在邮件服务器上，当以后用户重新连接 IMAP4 服务器时，这些邮件又可以与服务器进行再同步，显示全部的邮件内容。

（3）支持多个客户同时连接到一个邮箱。

POP3 协议在同一时间只能允许一个用户活动连接，而 IMAP4 协议允许同一时间不同的用户对同一邮箱的多个用户活动连接，并且提供了一种机制让客户能够感知当前连接到这个邮箱的其他用户的操作。这对于多个用户使用同一邮箱时非常有用，如一个项目组可能就使用同一个邮箱来进行邮件收发，不同项目成员可能会同时上网查看邮箱中各自所需的邮件。

（4）支持选择性获取。

如果你的 IMAP 客户端软件完整支持 IMAP4rev1 的话，则你还可以享受选择性下载附件的服务，也就是只下载正文或者部分或全部附件。例如在一封邮件里含有 3 个附件，但其中只有一个附件是你所需要的，这时你可以只下载那一个你所需要的附件，节省了下载其余两个附件的时间。

（5）支持在用户邮箱上创建、管理多个文件夹。

POP3 协议中，每个用户邮箱只能有几个默认的文件夹（如草稿箱、发件箱、收件箱、垃圾箱等），不可以新建其他文件夹，也不可以删除原来的这些默认文件夹，但在 IMAP4 服务器上每个默认文件夹下可以创建多个新的子文件夹，并且可以对各个邮箱文件夹进行重命名或删除操作。这样用户就可以对所接收或者发送的邮件分门别类地进行管理，就像本地磁盘上的文件夹管理一样。

（6）支持服务器端搜索。

IMAP4 协议提供了一种可以使客户端在服务器上搜索符合多个标准的信息的机制，就相当于在本地磁盘中进行文件搜索一样，IMAP4 的搜索功能更强大，不仅可以基于邮件标题，还可以基于邮件信封中的其他部分，甚至邮件的正文内容。这样，客户端就无需下载邮箱中的所有信息来完成这些搜索。

（7）支持客户端和服务器间的拖动操作。

一般的 IMAP4 客户端软件都支持邮件在本地文件夹间和服务器文件夹间的随意拖动，让你得心应手地把本地硬盘上的文件存放到服务器上，然后在你需要的时候同样方便地取回来，所有的功能仅需要用一次鼠标拖放的操作来实现，就像在网络中不同主机间的文件操作一样简单。

【说明】因为 IMAP4 协议相对来说更为复杂，而且目前 POP3 协议可以满足大多数电子邮件用户需求，所以在此不再对 IMAP4 协议进行详细介绍。

11.6 FTP 服务

FTP（File Transfer Protocol，文件传输协议）是一种主要的文件传输协议，经历了许许多多的版本更新，最新版本为 RFC 959。FTP 的主要功能是向用户提供本地和远程主机之间的文件传输，通过搭建 FTP 服务器（FTP 协议的服务器软件有很多，如 Windows 系统自带的 IIS 系统包括了 FTP 服务器功能、Serv-U 服务器、FileZilla 等）可实现文件资源共享，在网络设备上进行软件系统升级、日志下载和配置保存等业务操作时更是被广泛应用。

FTP 协议的优点是可以传输任意类型的文件，而不需要进行任何处理；缺点是因为在通过 FTP 协议进行文件传输时要先建立两种连接（控制连接和数据连接），所以有比较大的延时。

11.6.1 FTP 服务体系结构及传输方式

FTP 与大多数应用层协议一样也是采用 C/S 模式体系结构，如图 11-29 所示。

- FTP 服务器：运行于设备上的 FTP 服务提供远程 FTP 客户端访问和操作的功能，用户可以通过 FTP 客户端程序登录到 FTP 服务器上，访问 FTP 服务器上的文件资源。

- FTP 客户端：提供本地设备对远程 FTP 服务器上的文件进行操作的命令。用户在 PC 上通过终端程序或 Telnet 程序与作为 FTP 客户端的设备建立连接后，可以输入 FTP 命令建立与远程 FTP 服务器的连接并访问、操作远程 FTP 服务器上的文件。

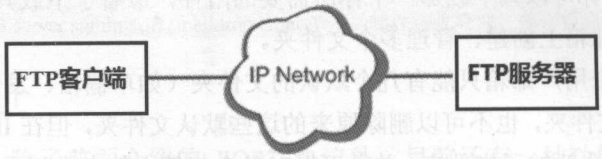


图 11-29　FTP 服务体系结构

担当 FTP 服务器的设备可以是计算机主机，也可以是像交换机、路由器等的网络设备。在计算机主机上需要安装所需的 FTP 服务器软件，在网络设备上一般是随设备操作系统自带的。至于 FTP 客户端功能，无论是计算机主机还是网络设备的操作系统中一般都自带有相应的 FTP 客户端程序和功能，无需另外安装软件。

在通过 FTP 协议进行文件传输时，命令的操作都是在 FTP 客户端进行的，FTP 服务器只提供相应的访问资源，当然 FTP 客户端既可以从 FTP 服务器上下载文件，也可以上传文件到 FTP 服务器上。有关 FTP 命令参见 11.6.7 节。

11.6.2　FTP 的两种工作模式

FTP 采用是面向连接的协议，所以它在进行文件传输前必须先在 FTP 服务器与 FTP 客户端之间建立好所需的 FTP 连接。但 FTP 协议的连接有比较特殊的一面，就是它包括“控制连接”和“数据连接”两种连接。其中控制连接用于连接控制端口，传输的是控制命令；数据连接用于实现数据传输。数据连接在控制连接建立之后，要通过控制端口的命令来建立数据连接，然后才能进行用户数据的传输。当控制连接建立后长时间没有建立数据连接时将自动中断。

在 FTP 数据连接的建立中又分为“主动模式”和“被动模式”两种，区别就在于“数据连接”是由 FTP 服务器先发起的还是由客户端先发起的，而控制连接都是由 FTP 客户端发起的。主动模式下的数据连接是由 FTP 服务器发起的，而被动模式下的数据连接是由 FTP 客户端发起的，此时 FTP 服务器处于被动监听状态。默认情况下采用主动模式，用户可以通过命令切换。

【注意】主动模式下，当 FTP 客户端存在防火墙时，由于数据连接是由 FTP 服务器主动向 FTP 客户端发起的，数据连接可能会发生问题。这是因为在主动模式下，除了 FTP 服务器要持续监听两个分别用于建立控制连接和数据连接的 TCP 端口外，FTP 客户端也需要持续监听一个用于建立数据连接的 TCP 端口（在本节后面将介绍），这样来自 FTP 服务器主动发起的数据连接请求有可能被 FTP 客户端的防火墙策略阻止了（防火墙的安全防护原则是“防外不防内”）。而在被动模式下，这个问题得到了解决，因为在被动模式下的数据连接是由 FTP 客户端主动发起的，此时 FTP 客户无需持续监听任何用于建立连接的 TCP 端口，只需 FTP 服务器持续监听两个分别用于建立控制连接和数据连接的 TCP 端口。

另外，FTP 连接的建立默认是无需认证的，但也可以出于安全考虑而配置认证功能，下面的连接建立流程是基于默认的匿名方式（无认证）进行介绍的。

1. 主动模式

主动模式下，两种 FTP 连接的建立过程所用到的 TCP 端口如图 11-30 所示。

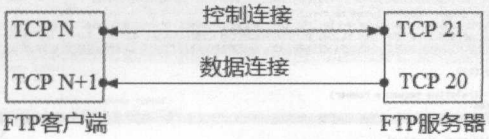


图 11-30 主动模式下两种 FTP 连接所用的端口

在主动模式下，FTP 的控制连接是由 FTP 客户端发起的（所使用的 TCP 端口是随机临时的，而且通常是大于 1024 的非知名服务所使用的端口，假设端口号为 N），FTP 服务器的控制连接端口默认是 TCP 21（可以修改）。在 FTP 客户端与 FTP 服务器之间建立好控制连接后，FTP 客户端通过 PORT 命令告知 FTP 服务器自己在建立数据连接时所用的 TCP 端口（也是大于 1024 的随机端口，且通常是在建立控制连接时所用的 TCP 端口号的基础上加 1，即 N+1），最后再由 FTP 服务器通过 TCP 20 端口主动向 FTP 客户端发起数据连接。具体流程如图 11-31 所示，文字描述如下：

（1）FTP 服务器打开 TCP 21 号端口，进入端口监听状态，等待 FTP 客户端发起的 FTP 控制连接的建立请求（可以同时与多个客户端建立连接）。

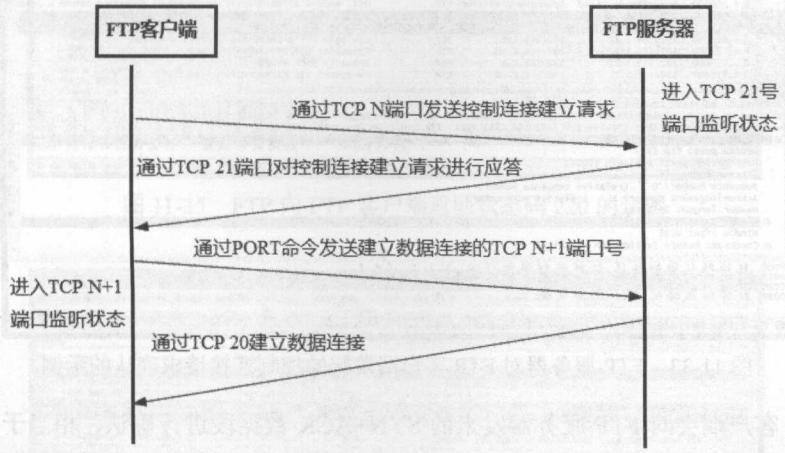


图 11-31 主动模式下两种 FTP 连接建立的流程

（2）当 FTP 客户端需要与 FTP 服务器交换文件时，通过一个大于 TCP 1024 号的非知名服务端口（假设端口号为 TCP N）向 FTP 服务器 TCP 21 端口发送 FTP 控制连接的建立请求，相当于 TCP 连接建立的第一次握手（发送的是 SYN 同步数据段），如图 11-32 所示（FTP 客户端 IP 地址为 192.168.1.10，FTP 服务器 IP 地址为 192.168.1.1，下同）。

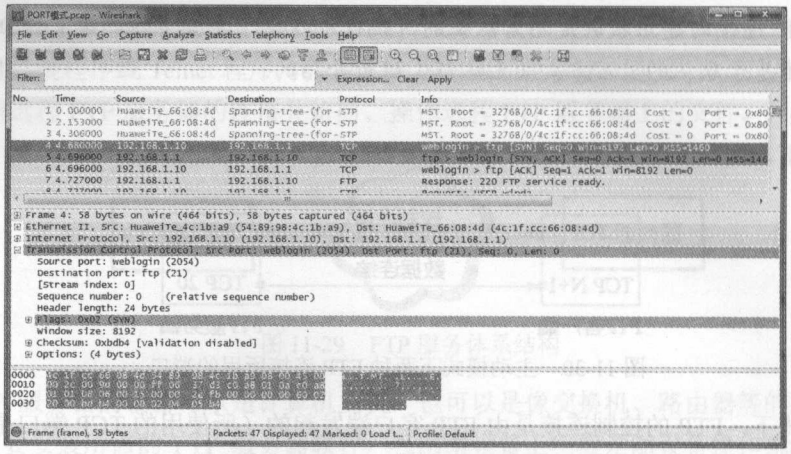


图 11-32　FTP 客户端向 FTP 服务器发送建立控制连接请求的示例

（3）FTP 服务器在收到 FTP 客户端发送的控制连接建立请求报文后，通过 TCP 21 号端口对 FTP 客户端的控制连接建立请求进行确认，相当于 TCP 连接建立的第二次握手（发送的是 SYN+ACK 数据段），如图 11-33 所示。

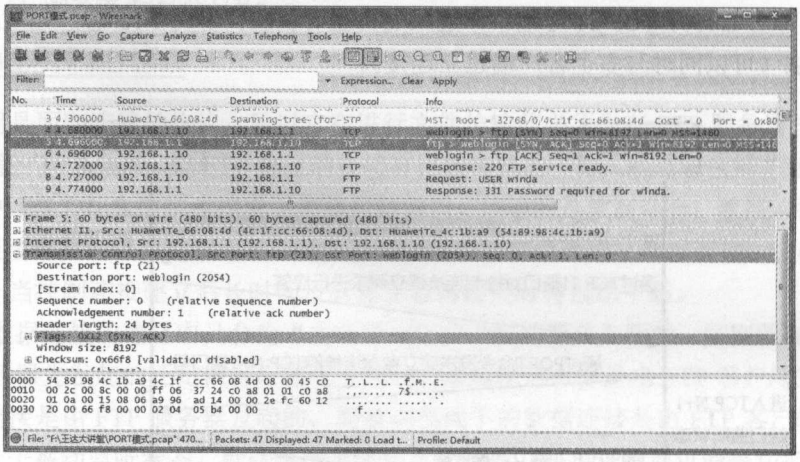


图 11-33　FTP 服务器对 FTP 客户端发起的控制连接请求确认的示例

此后，FTP 客户端会对 FTP 服务器发来的 SYN+ACK 数据段进行确认，相当于 TCP 连接建立的第三次握手，如图 11-34 所示。随后 FTP 服务器再向 FTP 客户端返回一个状态码为 220 的响应，表示控制连接建立成功，FTP 服务已正式开启，如图 11-35 所示。一个 FTP 控制连接仅用于一个对应 FTP 客户端与 FTP 服务器的连接。

（4）在 FTP 控制连接建立后，FTP 客户端向 FTP 服务器发送 PORT 命令，将它在数据连接建立时所使用的临时 TCP 端口号（通常是在 FTP 客户端在建立控制连接时所使用的临时端口号的基础上加 1，假设端口号为 TCP N+1）告诉 FTP 服务器，如图 11-36 所示。

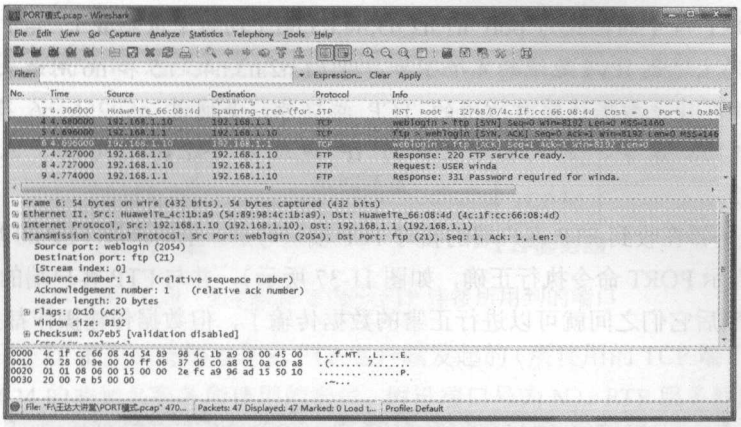


图 11-34　FTP 客户端对 FTP 服务器的 SYN+ACK 报文确认的示例

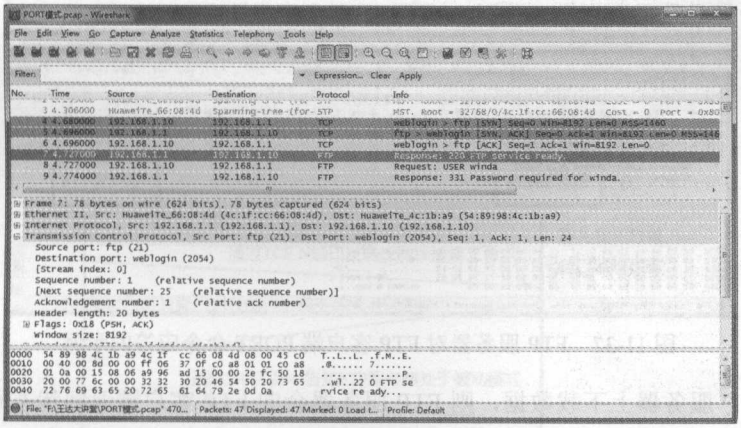


图 11-35　FTP 向 FTP 客户端返回状态码 220 的示例

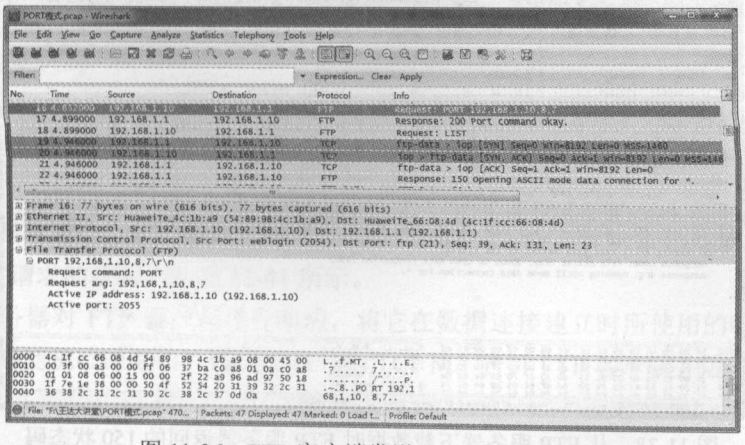


图 11-36　FTP 客户端发送 PORT 命令的示例

【说明】PORT 命令的格式为 port n1,n2,n3,n4,n5,n6（均为十进制数），其中 n1、n2、n3、n4 依次代表源主机的十进制 IPv4 地址的 4 个字节，即 n1.n2.n3.n4。n5 和 n6 用来表示要告知 FTP 服务器 FTP 客户端在建立数据连接时所用的 TCP 端口号，是通过 n5×256+n6 得到的。如 PORT 192,168,150,80,14,178 命令，则代表源主机的 IP 地址为 192.168.150.80，源 TCP 端口号为 14×256+178=3762。

（5）FTP 服务器在收到 FTP 客户端的临时 TCP 端口号后，主动通过 TCP 20 端口进行应答（返回状态码 200，表示 PORT 命令执行正确，如图 11-37 所示），并与 FTP 客户端的 TCP N+1 端口建立起数据连接。然后它们之间就可以进行正常的数据传输了，但数据传输操作都是在 FTP 客户端上进行的。

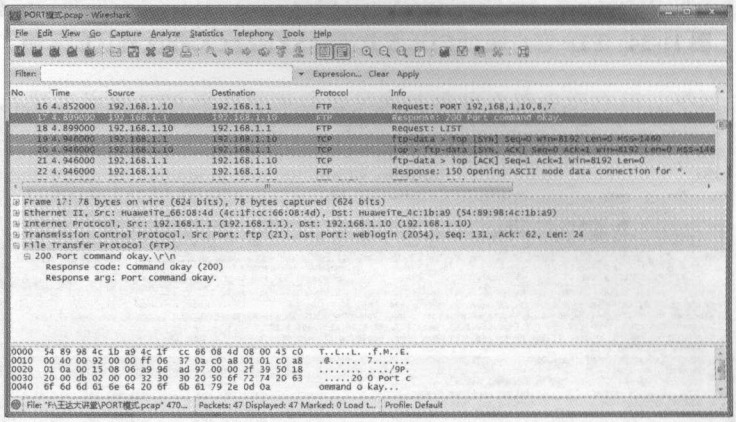


图 11-37　FTP 服务器对 FTP 客户端 PORT 命令应答的示例

如果要从 FTP 服务器上下载数据，则 FTP 服务器会向 FTP 客户端返回状态码 150，表示文件状态良好，可以进行数据传输，如图 11-38 所示。

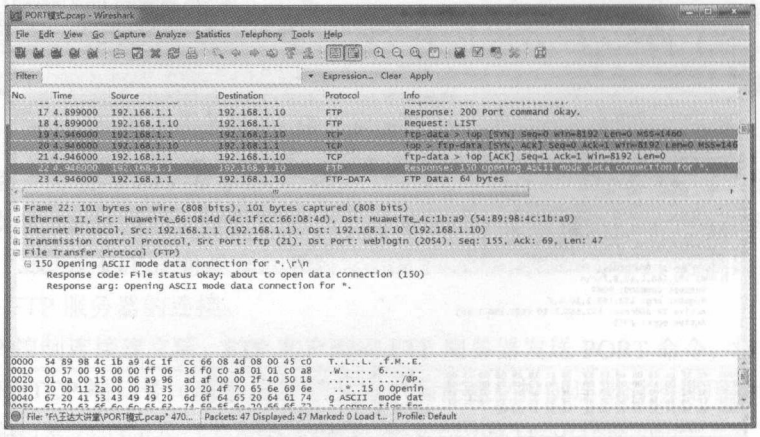


图 11-38　从 FTP 服务器下载数据时 FTP 服务器返回的 150 状态码

2. 被动模式

在被动模式下，两种 FTP 连接的建立过程所用到的 TCP 端口如图 11-39 所示。

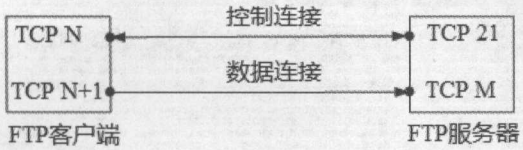


图 11-39 被动模式下两种 FTP 连接所用到的端口

在被动模式下，FTP 的控制连接也是由 FTP 客户端发起的（所使用的 TCP 端口是随机临时的，而且通常是大于 1024 的非知名服务所使用的端口，假设端口号为 N），FTP 服务器的控制连接端口默认也是 TCP 21。在 FTP 客户端与 FTP 服务器之间建立好控制连接后，再由 FTP 客户端通过控制连接发送 PASV 命令告知 FTP 服务器当前要处于被动模式，然后 FTP 服务器告知 FTP 客户端自己在建立数据连接时所使用的 TCP 端口（假设为 TCP M），最后再由 FTP 客户端向 FTP 服务器发起数据连接。具体流程如图 11-40 所示，文字描述如下（前 3 步建立控制连接的步骤是与主动模式是一样的）：

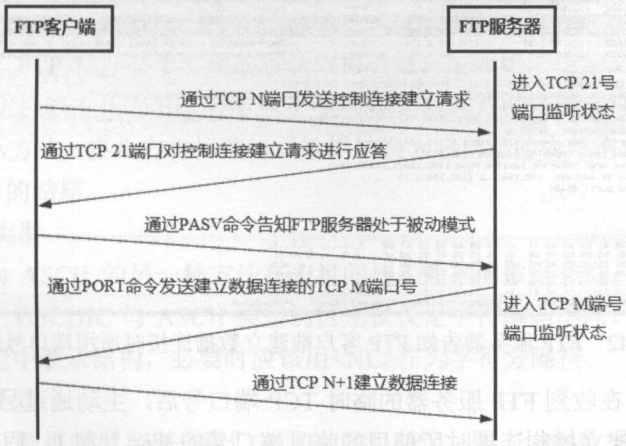


图 11-40 被动模式下两种 FTP 连接建立的流程

（1）～（3）参见“主动模式”中的（1）～（3）。

（4）FTP 客户端通过控制连接向 FTP 服务器发送命令 PASV，告知让 FTP 服务器处于被动模式，进入被动接收请求的状态，如图 11-41 所示。

（5）FTP 服务器对 FTP 客户端进行响应，将它在数据连接建立时所使用的临时 TCP 端口号（也是大于 1024 的随机临时端口，假设端口号为 TCP M）告诉 FTP 客户端并进入被动监听状态，如图 11-42 所示。

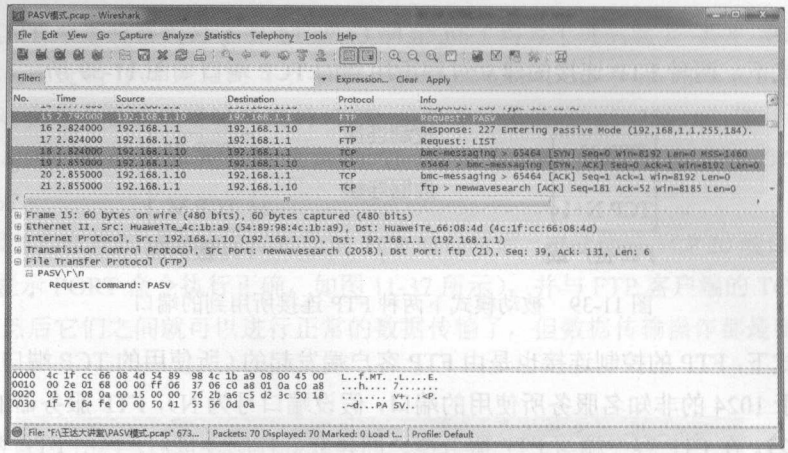


图 11-41　FTP 客户端发送 PASV 命令的示例

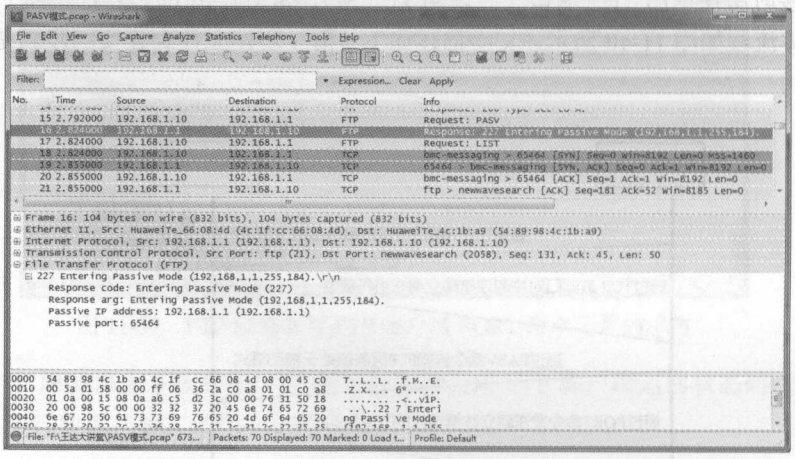


图 11-42　FTP 服务器告知 FTP 客户端建立数据连接时所用端口号的示例

（6）FTP 客户端在收到 FTP 服务器的临时 TCP 端口号后，主动通过另一随机 TCP 端口（通常是在 FTP 客户端在建立控制连接时所使用的临时端口号的基础上加 1，假设端口号为 TCP N+1）与 FTP 服务器端的 TCP M 端口建立数据连接。FTP 在收到连接请求后返回状态码 125，表示数据连接建立成功，可以进行数据传输了，如图 11-43 所示。然后它们之间就可以进行正常的数据传输了，但数据传输操作也都是在 FTP 客户端上进行的。

11.6.3　FTP 协议数据类型

FTP 协议的文件传输功能是通过数据连接实现的，控制连接仅用于操作命令的传输，在主机之间的数据传输中涉及到多个 FTP 命令。在数据传输中，涉及到 FTP 协议的数据表示类型、格式控制和数据结构等方面，本节和下面两节将分别予以介绍。

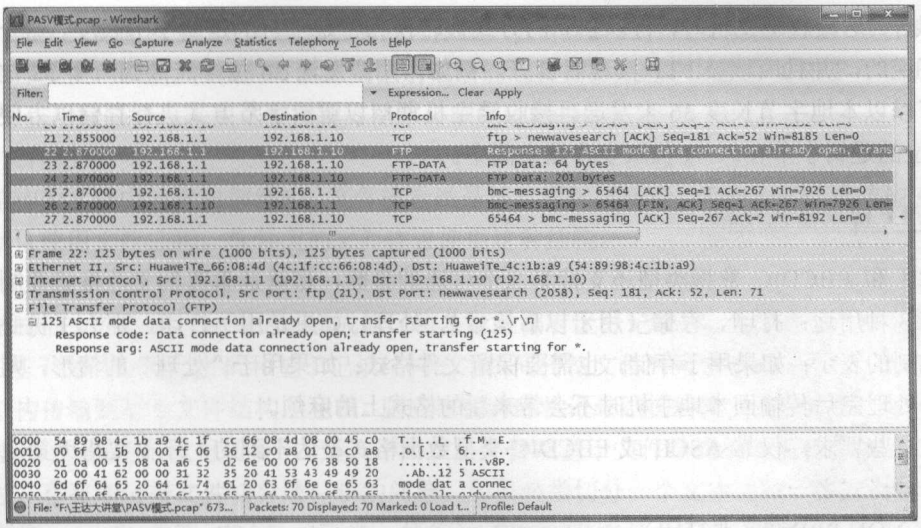


图 11-43 FTP 服务器向 FTP 客户端返回状态码 125 的示例

FTP 协议所支持的数据类型主要包括：ASCII、EBCDIC（Extended Binary Coded Decimal Interchange Code，扩展二进制编码的十进制交换码）、IMAGE（图像）和本地类型 4 种。

（1）ASCII 类型。

这是一种是所有 FTP 程序必须实现的默认数据类型，主要用来传输文本文件（包括各种文档类型的文件），除非主机双方认为 EBCDIC 类型更方便。发送方将内部字符表示方式转换为标准的 8 位 NVT-ASCII 表示方式，接收方将标准格式数据转换为它自己的内部格式，并且规定<CRLF>序列用来表示文本一行的结尾。

（2）EBCDIC 类型。

EBCDIC 是作为 ASCII 的另一种方法在主机间传送数据的数据类型，传输时数据被表示为 8 位的 EBCDIC 字符。EBCDIC 与 ASCII 类型的区别仅仅是字符编码不同。行尾（End-of-Line）很少用在 EBCDIC 类型中表示结构，必要时应该用<NL>作为字符分隔符。

（3）IMAGE 类型。

图像（IMAGE）数据类型（也是我们常见的 BINARY，即二进制数据类型）是以 8 位连续字节流传输，接收端也必须将数据存储为连续位。存储系统结构可能要将文件（或对于记录结构文件中的每个记录）填充到合适的边界（字节、字或块）。填充字节必须全为 0，并追加到文件末尾（或每个记录末尾）。必须有方法来指出填充字节，以便接收端在接收到文件时可以方便地将填充字节剔除。

（4）本地类型（Local Type）。

在“本地类型”中，数据是以参数 Bytesize（字节大小）指定的逻辑字节长度传输的。字节长度值必须是十进制整数，没有默认值，但逻辑字节长度不一定要和传输字节长度（也就是实际传输的数据长度）一样。如果不同，那么逻辑字节将忽略传输字节边界连续打包，并在最后做必要的填

充。当数据到达接收端主机时，将以独立的方式被转换为特定主机的逻辑字节长度。这个转换过程必须是可逆的，并且应该被 FTP 实现者公开。例如，用户发送 36 位浮点数到一个 32 位字长的主机时，可以以本地字节长度 36 来发送。接收端主机期望以逻辑字节方式进行存储以方便操作，但要将 36 位的逻辑字节存入 64 位双字存储空间中。

11.6.4 FTP 协议格式控制

ASCII 和 EBCDIC 数据类型还支持第二个可选的参数，那就是垂直的文件格式控制。字符文件可能有 3 种用途：打印、存储（用于以后检索）、处理。如果是用于打印，那么主机必须知道垂直格式控制的表示；如果用于存储，也需要保留文件格式；如果用于“处理”的情形，则要求在远程主机上处理完后传输回本地主机时不会带来新的格式上的麻烦。

以上这些要求，仅靠 ASCII 或 EBCDIC 类型数据格式是达不到的，还需要借助第二个参数指定以下 3 种格式之一：

（1）NON PRINT（非打印）。

当 ASCII 或 EBCDIC 类型文件中没有指定第二个参数时采用这种格式作为默认格式，不要包含垂直格式信息。如果文件被送到打印过程，那么打印过程将假定使用标准的间距和页边距值。一般来说，这种非打印格式主要用于处理或存储。

（2）TELNET 格式控制。

在这种格式的文件中包括 ASCII、EBCDIC 垂直格式控制字符，如<CR>（回车）、<LF>（行尾）、<NL>（换行）、<VT>（垂直制表）、<FF>（换页）和<CRLF>（回车换行，表示到达了行尾，编程时用“\r\n”表示），这些控制字符可以使打印机在打印过程中作适当的中断。

（3）CARRIAGE 控制（ASA）。

文件包含 ASA（FORTRAN）垂直格式控制字符。在以 ASA 标准形成的行中，第一个字符不打印，它用于决定打印前的走纸量。ASA 标准指定如下控制字符：

- 空：向下移动 1 行。
- 0：向下移动 2 行。
- 1：移动至下一页。
- +：不移动，即覆盖打印。

打印过程必须有方法区分结构体的结束。如果文件具有记录结构（本节后面将介绍）就没有问题，记录会在传输与存储中显式地标记。如果文件没有记录结构，将以<CRLF>行尾标记作为打印时行的分隔，但这些格式符会被 ASA 控制符覆盖。

11.6.5 FTP 协议数据结构

除了支持不同的数据类型外，FTP 协议还支持 3 种不同的数据结构：文件结构、记录结构和页结构。文件的结构会影响传输模型、存储和数据表示。文件本来的属性和保存它的主机有关，不同的机器会以自己的方式保存文件。在不同主机间传送文件时必须使主机能够识别相互的表示类型。

有些主机上的文件是面向字节的，有些是面向记录的，这时在传送时就会出现问题。那么就要在接收方进行内部转换。在进行转换的时候需要区别记录的边界，在 ASCII 中使用<CRLF>，在 EBCDIC 中使用<NL>作为分隔符。采用这种实现方法时必须保证转换是可逆的。

（1）文件结构。

这种结构的文件中没有内部结构，文件被看做是纯二进制比特流。如果没有使用结构（STRU）命令，则默认使用文件结构。软件类的文件建议使用这种结构。

（2）记录结构。

记录结构的文件是由一系列记录组成的（每条记录都有区分记录起、止的边界符），用于文本文件。但在基于文件结构的主机和基于记录结构的主机间传输文件时可能会出现问题。如果文件是基于记录结构传输到基于文件结构的主机，则应该在内部将记录结构转换为文件结构。显然这种转换应该能够可逆，以便可以再转换回记录结构。在将文件从基于文件结构的主机传输到基于记录结构的主机时，存在如何将文件切分成记录的问题。如果必须切分一个文本文件，那么 FTP 实现应该使用行尾符，ASCII 中是<CRLF>（回车、换行），EBCDIC 中是<NL>（换行）。

（3）页结构。

页结构的文件是由不同的索引页组成的（文件的一个部分称为页），每页都有页号和页头，可以进行随机存取和顺序存取。页头中有如表 11-17 所示的字段（所有字段都是一个逻辑字节，逻辑字节的长度由 TYPE 命令指定）。

表 11-17 页头字段

字段	含义
头长度	包括这个字节在内的头逻辑长度，最小头长度是 4
页索引	文件区域的逻辑页号，但并不是传输的序列号，而是标识本页的索引号
数据长度	页中数据的逻辑字节数，最小数据长度为 0
页类型	标示了页的类型，定义了如下类型： • 0=最末页：用来表示页结构传输结束，此时页头长度必须为 4，数据长度必须为 0 • 1=单独页：对于没有页相关控制信息的单独页来说这是普通的类型，此时页头长度必须为 4 • 2=描述页：用来传输整个文件的描述信息 • 3=存取控制页：包括一个额外地指定页存取信息的头域，此时头长度必须为 5
可选项	其他的头部字段，可以用来提供每页的控制信息，例如每页的存取控制

11.6.6 FTP 协议传输模式

利用 FTP 协议传输数据时还需要考虑选择合适的传输模式。FTP 协议支持 3 种传输模式：流模式、块模式和压缩模式。

1. 流模式

在“流模式”中，数据是以字节流方式进行传输的，是默认的传输模式。此时，对数据的表示类型没有限制，也可以使用记录结构。在记录结构文件中，EOR（记录结束）和 EOF（文件结束）

将分别用两个字节的控制码表示：第一个字节都是相同的 escape 字符，第二个字节中 EOR 将最低位置 1，其他位置 0，EOF 则是将第二低位置 1，也就是这个字节对于 EOR 来说是 1，对于 EOF 来说是 2。在传输结束时可以将 EOR 和 EOF 的最低两位同时置 1（就是值 3）。如果结构是文件结构，则使用关闭主机连接来指示 EOF，传输的所有数据字节就是原始字节，无需添加 EOR 和 EOF。

2. 块模式

在“块模式”中，文件是以连续的带有数据头的数据块来传输的。数据头一共 3 个字节，包括一个低 16 位的“字节计数器”字段和高 8 位的“描述符”，如图 11-44 所示。

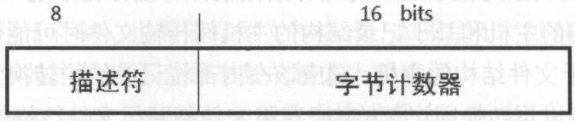


图 11-44　块模式下的“数据头”结构

“字节计数器”字段指示了数据块整个的长度，由此可以算出下一数据块的开始位置（没有填充位）。“描述符”字段则定义了：文件最后一块（EOF）、记录最后一块（EOR）、重开始标记（错误恢复和重开始）或者怀疑数据（也就是被怀疑在传输中可能不可靠的数据）。记录结构可以在这种模式下使用，而且可以用任何数据表示类型。

“描述符”字段由各个标志位组成，指定了 4 个描述码，每一个描述码为描述符的十进制值，具体如表 11-18 所示。

表 11-18　“描述符”字段取值的含义

描述码	含义
128	记录最后一块结束是 EOR
64	文件最后一块结束是 EOF
32	怀疑数据块有错
16	数据块是重开始标志。重开始标志是在数据流中的 8 位整数，表示在控制连接中使用的可打印字符（比如默认的 ASCII 字符），但不能使用<SP>（空格）

3. 压缩模式

压缩模式适用于在传输大数据时以较小的 CPU 代价换来一定的网络带宽。在压缩模式下有 3 种信息需要传送：①常规数据，以字节串传送；②压缩数据，包括数据副本和填充；③控制信息，以两个转义字符传送。

如果要发送 n（127>n>0）个字节的常规数据，则在该数据的最高位前加一个头字节，其中最高位为 0，其余位表示的数值等于 n，如图 11-45 所示。

为了压缩 n 个字节的数据副本，需要发送如图 11-46 所示的两个字节。最高位固定为 10，后面 6 位的 n 代表压缩的字节数，最低的 8 位用来表示 n 个字压缩后的一个字节数据。

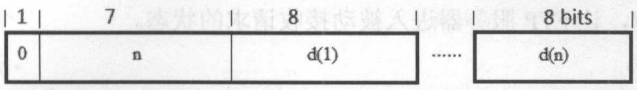


图 11-45 压缩模式发送常规数据的格式

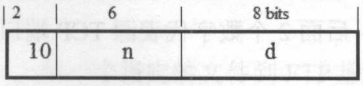


图 11-46 压缩模式发送压缩数据时的“数据副本”格式

紧接数据副本的是“填充字节”，其格式如图 11-47 所示（代表每一个压缩单元在接收端恢复时都要用填充字节还原），其中高 2 位固定为 11，低 6 位的 n 代表具体的填充值。填充值与数据表示类型有关：如果类型是 ASCII 或 EBCDIC，填充字节是<SP>（空格，ASCII 码是 32，EBCDIC 码是 64）；如果类型是图像或本地字节，填充字节为 0。

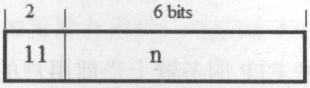


图 11-47 压缩模式发送压缩数据时的“填充”字节格式

在“填充字节”后面是“转义字符”，它是由两个字节组成的，第一个字节为转义字节（全 0），第二个字节包括“块模式”中定义的描述码。这里的描述码与块模式中的描述码意义相同，作用于其后的字节串。

11.6.7 FTP 协议操作命令

在 FTP 协议客户端可以操作的命令非常多，这些命令总的来说可以分为：访问控制命令、模式设置命令、文件管理命令、文件传输命令和其他命令 5 类。

1. 访问控制命令

（1）USER 命令。

命令格式：USER username，指定进行 FTP 文件访问的用户账户名，以便 FTP 服务器对 FTP 客户端进行身份验证。参数 username 是具体的用户账户名。

（2）PASS 命令。

命令格式：PASS password，指定进行 FTP 文件访问的用户账户密码。参数 password 是具体的用户账户密码，该命令必须跟在 USER 命令之后。

（3）ACCT 命令。

命令格式：ACCT username password，指定进行 FTP 文件访问的用户账户（包括用户名和密码）。

（4）QUIT 命令。

命令格式：QUIT，关闭与 FTP 服务器的控制连接，终止当前用户访问 FTP 服务器。

2. 模式设置命令

（1）PASV 命令。

命令格式：PASV，让 FTP 服务器进入被动接收请求的状态。

（2）PORT 命令。

命令格式：PORT address，定义数据连接所使用的源 IPv4 地址和源 TCP 端口。参数 address 包括 6 个用逗号分隔的十进制数字（如 n1、n2、n3、n4、n5、n6），前面 4 个数字分别代表源 IPv4 地址的 4 个字节（从高到低）值，后面 2 个数字代表源 TCP 端口，等于 n5×256+n6。

（3）TYPE 命令。

命令格式：TYPE datatype，指定要传输的数据类型，有 ASCII、BINARY（二进制）/IMAGE（图像）、EBCDIC、Telnet 和 Nonprint（非打印），分别用 A、I、E、T 和 N 表示。

（4）MODE 命令。

命令格式：MODE mode，指定传输方式。参数 mode 的取值有：S（表示流方式）、B（表示块方式）、C（表示压缩方式）。

3. 文件管理命令

（1）CWD 命令。

命令格式：CWD directory，改变 FTP 服务器上当前用户的主目录。

（2）PWD 命令。

命令格式：PWD，显示 FTP 服务器上当前用户的主目录。

（3）MKD 命令。

命令格式：MKD directory，在指定路径下创建指定的新目录。

（4）CDUP 命令。

命令格式：CDUP，返回到上层目录。

（5）RMD 命令。

命令格式：RMD directory，删除指定目录。

（6）LIST 命令。

命令格式：LIST [name]，返回指定路径下的子目录及文件列表，省略 name 参数时返回当前路径下的文件列表。

（7）NLST 命令。

命令格式：NLST [directory]，返回指定路径下的目录列表，省略 directory 参数时返回当前目录。

（8）RNFR 命令。

命令格式：RNFR old path，重新命名文件，该命令的下一条命令应该用 RNTO 指定新的文件名。

（9）RNTO 命令。

命令格式：RNTO new path，该命令和 RNFR 命令共同完成对文件的重命名，紧跟在 RNFR 命令之后。

（10）DELE 命令。

命令格式：DELE filename，删除指定路径下的文件。参数 filename 包括要删除文件的完整路径。

4. 文件传输命令

（1）RETR 命令。

命令格式：RETR filename，请求 FTP 服务器将指定路径内的文件复制到客户端，亦即下载指定的文件。参数 filename 包括要下载文件的完整路径。

（2）STOR 命令。

命令格式：STOR filename，上传一个指定的文件到 FTP 服务器，并将其存储在指定的位置。参数 filename 包括要上传文件的完整路径。如果指定路径下已存在同名文件，则覆盖它。

（3）APPE 命令。

命令格式：APPE filename，与 STOR 命令类似，但使用本命令上传文件到 FTP 服务器时，如果指定路径下已有同名文件，则附加在原文件之后。

（4）ABOR 命令。

命令格式：ABOR，通知服务终止以前的 FTP 命令和与之相关的数据传送。如果先前的操作已经完成，则没有动作，返回 226；如果没有完成，则返回 426，然后再返回 226。关闭控制连接，但数据连接不关闭。

（5）DELE 命令。

命令格式：DELE filename，删除指定路径下的指定文件。

5. 其他命令

（1）SYST 命令。

命令格式：SYST，返回 FTP 服务器上使用的操作系统。

（2）STAT 命令。

命令格式：STAT，返回 FTP 服务器上当前的控制连接状态。

（3）REST 命令。

命令格式：REST offset，由特定偏移量开始重启文件传输进程。参数 offset 用来指定重传的开始字节偏移量。

（4）HELP 命令。

命令格式：HELP，获取 FTP 系统帮助。

11.6.8 FTP 协议响应码

在执行上节的 FTP 命令时，根据最终的执行结果会返回相应的代码。这些代码代表了不同的含义，具体如表 11-19 所示。

表 11-19 FTP 协议响应码

响应码	含义
110	重新启动标记应答，标记格式为 MARK yyyy=mmmm，其中 yyyy 是用户进程数据流标记，mmmm 是服务器标记

续表

响应码	含义
120	服务器准备就绪的时间（分钟数）
125	数据连接已打开，准备进行数据传输
150	文件状态良好，打开数据连接
200	命令执行成功
202	命令没有执行
211	系统状态应答
212	目录状态应答
213	文件状态应答
214	帮助信息应答
215	系统类型应答
220	服务就绪
221	关闭控制连接，可以退出网络
225	数据连接打开，无数据传输
226	关闭数据连接，文件传输完成
227	进入被动模式（IP 地址、TCP 端口）
230	登录成功
250	请求的文件操作完成
257	路径名建立
331	用户名正确，要求密码
332	要求账号信息
350	请求的文件操作需要进一步的命令
421	不能提供服务，关闭控制连接
425	无法打开数据连接
426	关闭连接，终止传输
450	请求的文件操作未执行
451	终止请求的操作：有本地错误
452	未执行请求的操作：磁盘空间不足
500	格式错误：命令不可识别
501	参数语法错误
502	命令没有执行
503	命令顺序错误

续表

响应码	含义
504	无效命令参数
530	未登录网络
532	存储文件需要账号信息
550	未执行请求的操作
551	请求操作终止：页面类型未知
552	请求操作终止：存储分配溢出
553	请求操作终止：文件名不合法

11.6.9 FTP 协议身份验证

在 FTP 客户端与 FTP 服务器进行控制连接建立时，除了可以采用默认的匿名方式外，还可以配置身份验证，只有在通过身份验证后才能进行控制连接的建立。这时就需要在 FTP 服务器上配置相应的用户账户名和密码。另外，还可以为不同用户创建并指定一个工作主目录，这样当 FTP 用户从FTP客户端访问FTP服务器时就自动进入到为该用户指定的用户主目录下的子目录和文件，而且该用户只能在指定的主目录下进行访问，不能访问其他目录及其下的子目录和文件。

如图 11-48 所示是当 FTP 客户端与 FTP 服务器建立控制连接时的用户身份验证以及主目录指定的流程图。有关 FTP 服务器上的身份验证和用户主目录配置参见相应的 FTP 服务器软件说明，网络设备上的 FTP 服务器配置参见《Cisco/H3C 交换机配置与管理完全手册》（第三版）和《华为交换机学习指南》。

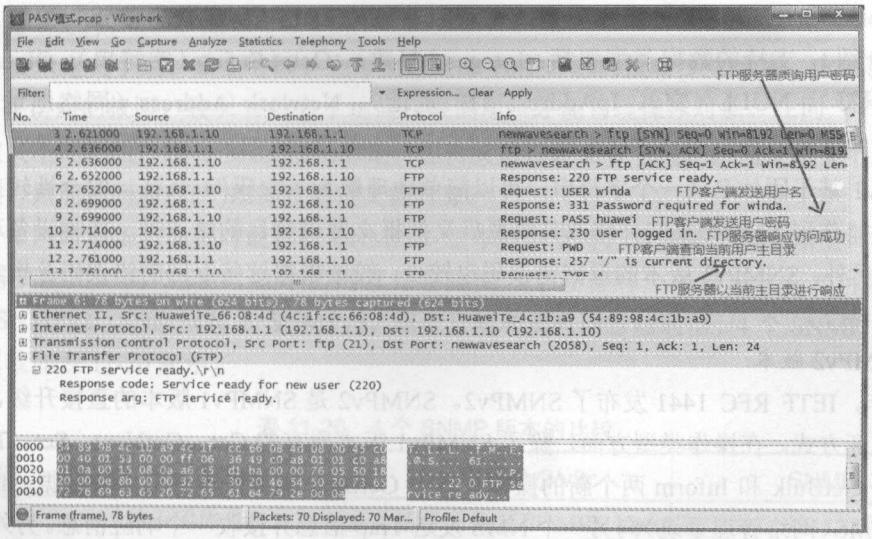


图 11-48 FTP 身份验证流程示例

11.7 SNMP服务

我们在了解网络设备（如交换机、路由器等）支持的管理方式中经常见到说支持SNMP管理，这就是指该设备可以通过安装了SNMP（Simple Network Management Protocol）软件系统的工作站进行配置与管理。一般情况下，SNMP管理方式不同于我们常见的CLI（命令行接口）管理方式的shell界面，而通常是图形界面方式，操作更加简单。

通过SNMP协议进行网络设备管理时，是通过一系列的操作（如后面将要介绍的Get、GetNext、Set、Response和Trap等），亦即对应的一系列PDU或消息来实现管理站与设备之间的信息交互的，最终达到对网络设备的被管理对象属性读取和写入的目的。通过SNMP协议进行网络设备管理的最大优势就是因为它是通用协议，所以屏蔽了不同产品之间的差异，实现了不同种类和厂商的网络设备之间的统一管理。即采用SNMP协议开发的网管系统不同于一些特定厂商设备的网管系统，它能管理所有支持SNMP管理的设备，不限于某一个品牌。如华为基于SNMP协议开发的eSight系统不仅可以管理华为的交换机、路由器和防火墙产品，还可以管理Cisco和H3C等品牌的网络设备，而像CiscoWorks系统就只能管理Cisco的设备。

11.7.1 SNMP协议的主要版本

SNMP与其他所有协议一样也在不断发展之中，所以也出现过多个不同的版本。目前，各大品牌网络设备所使用的SNMP协议主要是：SNMPv1、SNMPv2、SNMPv2c和SNMPv3这4个版本。

1. SNMPv1版本

1990年5月，IETF RFC 1157定义了SNMP的第一个版本SNMPv1，支持Get、GetNext、Set、Response和Trap这5种操作类型，也对应着5种PDU（协议数据单元）类型（具体将在本章后面介绍），支持的数据类型包括：Integer（整数）、Octet String（字符串）、Object Identifier（对象标识符）、NULL（空）、IpAddress（IP地址）、Network Address（网络地址）、Counter（计数器）等。

SNMPv1基于团体名（一个字符串，可以简单地理解为“密码”）认证，安全性较差，未经验证的用户在捕获到SNMP报文并修改其信息后又把报文发送到目的站点时，接收设备不能获知数据的改变。另外，SNMPv1版本的返回错误码也较少，较难根据这些返回的错误码分析具体的故障原因。

2. SNMPv2版本

1993年，IETF RFC 1441发布了SNMPv2。SNMPv2是SNMPv1版本的直接升级，但仍采用基于团体认证方式。在操作类型方面，除了SNMPv1版本原有的Get、GetNext、Set、Trap等操作外，增加了GetBulk和Inform两个新的操作。其中GetBulk操作快速获取大块数据，Inform操作允许一个NMS（网络管理系统）向另一个NMS发送Trap信息并接收一个响应消息。另外，SNMPv2版本添加了几个新的数据类型（Counter32、Counter64、Gauge32、UInteger32、NsapAdress和BIT

String），并对 OID 表和 OID 值的设置增强了。

在安全性方面，SNMPv2 版本对数据修改、假冒和数据包顺序改变等安全问题提出了比较满意的解决方案，同时支持分级的安全机制和验证机制，并使用 DES（数据加密标准）加密算法，安全性远比 SNMPv1 版本高。但由于它的子类比较多，其提供的安全性功能实现起来比较复杂，现在比较少用了，更多的是采用它的其中一个子类——SNMPv2c。

3. SNMPv2c 版本

尽管 SNMPv2 版本提供了许多高级安全功能，但却并没有得到厂商的积极响应，于是 1996 年 IETF 在 RFC 1901 中发布了一个简化的 SNMPv2 版本——SNMPv2c，这个 c 就是 community-based（基于团体）的意思，删除了原来 SNMPv2 版本中的这些安全功能，但保留了 SNMPv2 版本中新增的操作类型和新支持的数据类型，也采用基于团体名的认证方式；提供了更丰富的错误代码，能够更细致地区分错误。

4. SNMPv3

1998 年 IETF RFC 2271（还有许多配套的 RFC 文档）发布了 SNMPv3 版本，主要是在 SNMPv2 基础之上增加、完善了安全和管理机制，并且把前面版本中的代理进程（Agent）称为“实体”（Enty），除了新增了 Report PDU 外其他 PDU 类型与 SNMPv2 版本一样，并且同时兼容 v1、v2 和 v2c 版本。在安全性方面，SNMPv3 版本提供了基于 USM（User Security Module）的认证加密和基于 VACM（View-based Access Control Model）的访问控制，不再基于团体名认证，不仅在安全性方面得到了显著提高，而且实现起来也比较容易。

USM 引入了用户名和组的概念，可以为单个用户或一组用户设置认证和加密功能。认证用于验证报文发送方的合法性，避免非法用户的访问；加密则是对 NMS 和 Agent 之间传输的报文进行加密，以免被窃听。通过有无认证和有无加密等功能组合，可以为 NMS 和 Agent 之间的通信提供灵活的安全选项支持。

VACM 技术定义了组、安全等级、上下文、MIB 视图、访问策略 5 个元素，它们同时决定了用户是否具有访问的权限，使只有具有了访问权限的用户才能管理操作对象。在同一个 SNMP 实体上可以定义不同的组，组与 MIB 视图绑定，组内又可以定义多个用户。当使用某个用户名进行访问的时候，只能访问对应的 MIB 视图定义的对象。

目前许多网络设备仍没有对最新的 SNMPv3 版本提供支持，特别是一些早期上市的网络设备产品。当需要使用不同 SNMP 版本的网管站管理同一个设备时，在该设备上尽可能同时启用 SNMPv1、SNMPv2、SNMPv2c 和 SNMPv3，保证它能和所有网管站通信。4 个 SNMP 协议版本的主要特性比较如表 11-20 所示。

表 11-20 4 个 SNMP 版本的比较

特性	SNMPv1	SNMPv2/v2c	SNMPv3
访问控制	基于团体名和 MIB View 进行访问控制	基于团体名和 MIB View 进行访问控制	基于用户、用户组和 MIB View 进行访问控制

续表

特性	SNMPv1	SNMPv2/v2c	SNMPv3
认证加密	基于团体名的认证	基于团体名的认证 SNMPv2 版本支持分级认证和 DES 加密	支持认证和加密： 认证：MD5、SHA 加密：DES56、AES128
错误码	支持 6 个错误码	支持 16 个错误码	支持 16 个错误码
Trap 告警	支持	支持	支持
GetBulk 和 Inform 操作	不支持	支持	支持

11.7.2 SNMP 协议管理模型

整个 SNMP 系统也是采用 C/S 结构，管理员操作的、运行 SNMP 协议的网络管理软件主机属于客户端，而被管理的设备属于服务器。一套基于 SNMP 协议的网管系统由网络管理系统（Network Management System，NMS）、代理进程（Agent）、管理对象（Management Object）和管理信息库（Management Information Base，MIB）四部分组成，如图 11-49 所示。

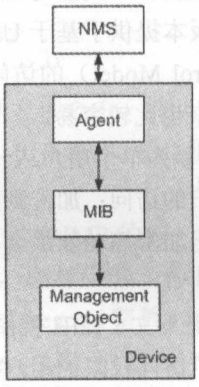


图 11-49　SNMP 网管系统的基本组成

NMS 作为整个网络的网管中心，对网络中的设备进行管理。每个被管理设备中都包含驻留在设备上的 Agent 进程、MIB 和多个被管理对象。NMS 通过与运行在被管理设备上的 Agent 交互，由 Agent 通过对设备端的 MIB 的操作，完成 NMS 的指令。

1. NMS

- NMS 在网络中扮演管理者角色，是一个采用 SNMP 协议对网络设备进行管理/监视的系统，运行在 NMS 服务器上。
- NMS 可以向设备上的 Agent 发出请求，查询（通过 Get 或 GetNext 操作）或修改（通过 Set 操作）一个或多个具体的参数值。
- NMS 可以接收设备上的 Agent 主动发送的 Trap 信息，以获知被管理设备当前的状态。

2. Agent

- Agent 是被管理设备中的一个代理进程，用于维护被管理设备的信息数据并响应来自 NMS 的请求，把管理数据汇报给发送请求的 NMS。
- Agent 接收到 NMS 的请求（写或读操作）信息后，通过 MIB 表完成相应指令并把操作结果响应给 NMS。
- 当设备发生故障或者其他事件时，设备会通过 Agent 主动发送 Trap 信息给 NMS，向 NMS 报告设备当前的状态变化。

3. Management Object

每一个设备可能包含多个 Management Object（被管理对象）。被管理对象可以是设备中的某个硬件（如一块接口板、一个主控板等）、软件功能（如一个 VLAN、一个 IP 地址等），也可以是某些硬件、软件（如路由选择协议）及其配置参数的集合。

4. MIB

MIB 是一个数据库，指明了被管理设备所维护的变量（即能够被 Agent 查询和设置的信息）。MIB 在数据库中定义了被管理设备的一系列属性，包括对象名、对象状态、对象的访问权限和对象的数据类型等。通过 MIB 可以完成以下功能：

- Agent 通过查询 MIB 可以获知被管理对象当前的状态信息。
- Agent 通过修改 MIB 可以设置被管理对象的状态参数。

SNMP 体系结构中的 MIB 为对象命名树，采用和 DNS（域名系统）相似的树状结构，如图 11-50 所示。它的根在最上面，根没有名字，树的节点表示被管理对象，每个对象既可以用名称（如图中的 ccit、iso、org 等）进行标识，也可以用一个在同一层次中唯一的数字（如对应的 0、1、3 等）进行标识（不同层的对象名和标识可以一样）。这也像 DNS 域名系统要求在一个域名树中同级域名的名称必须保持唯一一样。

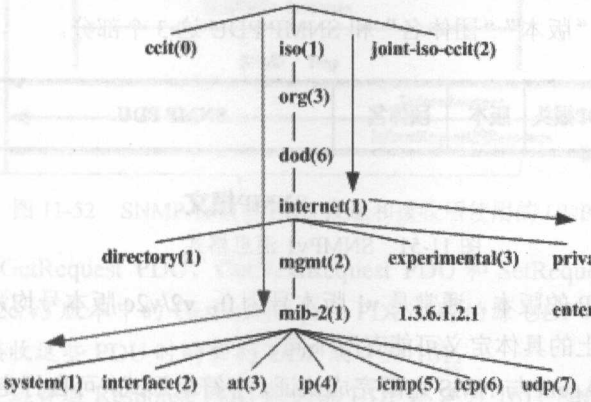


图 11-50 MIB 树状结构

每个被管理对象在 SNMP 管理中不是用该对象上的数字来进行标识，而是用一个从根开始到

达该对象的数字标识串进行唯一标识，就像一个域名不是由某一级域名标识，而是由一个从根开始的完整域名进行标识一样。即被管理对象可以用从根开始的一条路径唯一地识别，这条路径就称为OID（Object IDentifier，对象标识符）。在图 11-50 中被管理对象 system 可以用一串数字{1.3.6.1.2.1.1}唯一标识，这串数字就是 system 的 OID。子树可以用该子树根节点的 OID 来标识，如图中以 private 为根节点的子树的 OID 为 private 的 OID——{1.3.6.1.4}。

子树掩码可以和子树 OID 共同来确定一个视图的范围。子树掩码用十六进制格式表示，转化成二进制后，每个比特位对应 OID 中的一个小节，其中：

- 1 表示精确匹配，即要访问的节点 OID 与 MIB 对象子树 OID 对应小节的值必须相等。
- 0 表示通配，即要访问的节点 OID 与 MIB 对象子树 OID 对应小节的值可以不相等。

例如，子树掩码为 0xDB（二进制格式为 11011011），子树 OID 为 1.3.6.1.6.1.2.1，则所确定的视图包括子树 OID 为 1.3.*.1.6.*.2.1（*表示可为任意数字）的子树下的所有节点。

【说明】若子树掩码的位数大于子树 OID 的小节数，则匹配时，子树掩码的第一位与子树 OID 的第一小节对齐，第二位与第二小节对齐，依此类推，子树掩码中多出的位将被忽略；若子树掩码的位数小于子树 OID 的小节数，则匹配时，子树掩码的第一位与子树 OID 的第一小节对齐，第二位与第二小节对齐，依此类推，子树掩码中不足的位将自动设置为 1；如果没有指定子树掩码，则使用默认的子树掩码（全 1）。

11.7.3 SNMPv1 报文格式

通过前面的介绍已经知道，当前 SNMP 协议主要有 v1、v2、v2c 和 v3 这 4 个版本，但 SNMPv2 版本由于过于复杂现在很少使用了，本节及后面两节仅介绍 SNMPv1、v2c 和 v3 这 3 个版本的报文格式。

SNMP 是应用层协议，而且采用的是 UDP 传输层协议进行传输，在 IP 网络中当 SNMP 消息到达网络层后还要经过 IP 协议封装。在网络层 SNMPv1 的 IP 数据报格式如图 11-51 所示，其中属于 SNMP 报文的部分包括“版本”“团体名”和 SNMP PDU 这 3 个部分。

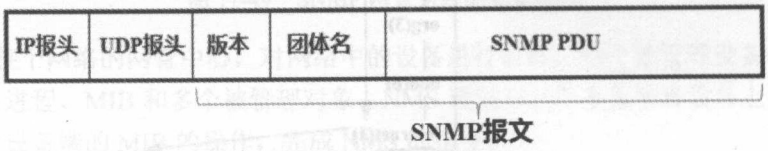


图 11-51 SNMPv1 消息格式

- 版本：表示 SNMP 的版本。通常是 v1 版本号为 0，v2/v2c 版本号均为 1，v3 版本号为 2。但不同厂商设备上的具体定义可能有所不同。
- 团体名：用于在 Agent 与 NMS 之间完成认证，字符串形式，可自行定义。团体名包括“可读”和“可写”两种（区分读取和写入权限，更加安全），如果执行 GetRequest、GetNextRequest 操作时，采用可读团体名进行认证；如果执行 Set 操作时，采用可写团体

名认证。如果认证失败，SNMP 协议的代理将向管理站发送一个认证失败的 Trap 消息。

- SNMP PDU：包含 PDU 类型、请求标识符（请求 ID）、变量绑定等信息。SNMPv1 包括的 PDU 类型有：GetRequest、GetNextRequest、SetRequest、Response 和 Trap，它们的作用如表 11-21 所示。

表 11-21 SNMPv1 版本各种 PDU 的作用

PDU 类型	描述
GetRequest	GetRequest PDU 可以从 Agent 中提取一个或多个参数值
GetNextRequest	GetNextRequest PDU 可以从 Agent 中按照字典序提取下一个参数值
SetRequest	SetRequest PDU 可以设置 Agent 的一个或多个参数值
Response	Response PDU 可以返回一个或多个参数值。这个操作是由 Agent 发出的，是 GetRequest、GetNextRequest、SetRequest 和 GetBulkRequest 四种 PDU 操作的响应操作。Agent 接收到来自 NMS 的 Get/Set 指令后，通过 MIB 完成相应的查询/修改操作，然后利用 Response PDU 将信息回应给 NMS
Trap	Trap PDU 信息是 Agent 主动向 NMS 发出的信息，告知管理进程设备端出现的情况

【说明】SNMP 协议的各种 PDU 发送和接收所使用的 UDP 端口有所不同，如图 11-52 所示，具体描述如下：

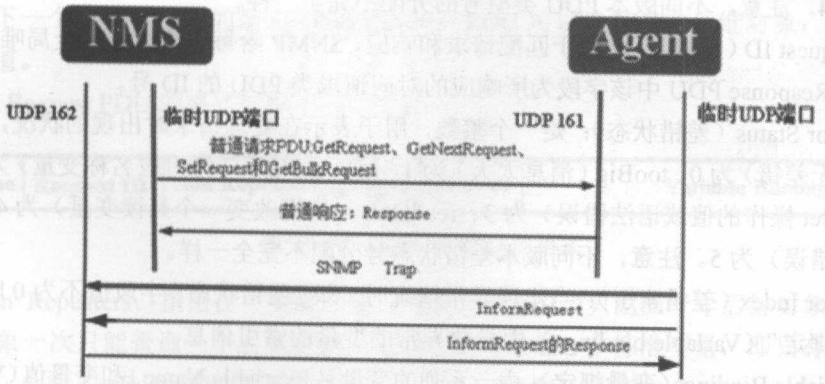


图 11-52 SNMP 协议各 PDU 发送和接收所使用的 UDP 端口

- NMS 发送 GetRequest PDU、GetNextRequest PDU 和 SetRequest PDU（包括后面介绍的 SNMPv2/v2c/v3 版本中的 GetBulkRequest PDU）时的源 UDP 端口是临时、动态分配的，代理进程接收这些 PDU 时的目的 UDP 端口为 161。
- 代理进程发送普通 Response PDU 时的源 UDP 端口为 161，NMS 接收普通 Response PDU 的目的 UDP 端口为临时、动态的端口（与请求 PDU 发送时的临时端口一致）。
- 代理进行发送 Trap PDU 和 InformRequest PDU 时的源 UDP 端口为临时、动态的，NMS

接收这些 PDU 的目的 UDP 端口为 162。

GetRequest PDU、GetNextRequest PDU、SetRequest PDU 的格式如图 11-53 所示，Response PDU 的格式如图 11-54 所示，Trap PDU 的格式如图 11-55 所示。

PDU Type	Request ID	0	0	Variable Bindings

图 11-53　GetRequest/GetNextRequest/SetRequest PDU 格式

PDU Type	Request ID	Error Status	Error Index	Variable Bindings

图 11-54　Response PDU 格式

PDU Type	Enterprise	Agent Addr	Generic Trap	Specific Trap	Time Stamp	Variable Bindings

图 11-55　Trap PDU 格式

以上 PDU 格式中各字段的含义如下：

- PDU Type（PDU 类型）：在 SNMPv1 中，为各种 PDU 类型分配了一个 ID 号：GetRequest PDU 为 0，GetNextRequest PDU 为 1，Response PDU 为 2，SetRequest PDU 为 3，Trap PDU 为 4。注意，不同版本 PDU 类型号的分配不完全一样。
- Request ID（请求 ID）：用于匹配请求和响应，SNMP 给每个请求分配全局唯一的请求 ID。在 Response PDU 中该字段为所响应的对应请求类 PDU 的 ID 号。
- Error Status（差错状态）：是一个整数，用于表示在处理请求时出现的状况，包括 noError（无差错）为 0、tooBig（消息太大）为 1、noSuchName（无对应名称变量）为 2、badValue（Set 操作的值或语法错误）为 3、readOnly（试图改变一个只读变量）为 4、genErr（其他错误）为 5。注意，不同版本差错状态号分配不完全一样。
- Error Index（差错索引）：当出现异常情况时（即“差错状态”字段值不为 0 时），提供“变量绑定”（Variable bindings）中导致异常的变量的索引信息。
- Variable Bindings（变量绑定）：由一系列的变量名（Variable Name）和变量值（Variable Value）对组成。对于 GetRequest PDU、GetNextRequest PDU 是请求提取相应变量的参数；对于 SetRequest PDU 是请求设置相应变量的参数；对于 Response PDU 是指明一个或多个变量的名和对应的值；对于 Trap PDU 是返回的 Trap 消息中涉及的 OID 及相应错误消息。
- Enterprise：Trap PDU 中特有，代表生成此 Trap 信息的设备名称。
- Agent Addr（代理地址）：Trap PDU 中特有，是 Trap 源设备的 IP 地址。
- Generic Trap（通用 Trap）：Trap PDU 中特有，包括 coldStart（冷启动）、warmStart（热启动）、linkDown（链路关闭）、linkUp（链路打开）、authenticationFailure（认证失败）、

egpNeighborLoss（外部邻居丢失）和 enterpriseSpecific（设备参数），取值分别为 0～6。

- Specific Trap（特定 Trap）：Trap PDU 中特有，可以与“通用 Trap”一样，也可以是企业自己指定的 Trap 类型。
- Time Stamp（时间戳）：Trap PDU 中特有，指从上次重新初始化对象到该对象产生此 Trap 消息所经过的时间，即 sysUpTime（系统正常时间）对象的取值。

11.7.4 SNMPv2c 报文格式

SNMPv2c 消息的总体结构与 SNMPv1 一样，参见图 11-51。在 PDU 方面，比较 SNMPv1 而言主要改进体现在两个方面：一是对原有 PDU 工作机制进行了改进，即在 GetRequesst PDU 和 GetNextRequest PDU 中均允许部分应答，而不是像在 SNMPv1 版本中，只要有一个变量取值没有获得所有变量值都不能返回，提高了操作的效率。另外，SNMPv2c 新增了 GetBulkRequest PDU（类型号为 5），实现了 NMS 对被管理设备的信息群查询（一次查询多个信息），修改了 Trap PDU 格式（在 SNMPv2 中 Trap PDU 的 PDU 类型 ID 号为 7）。

1. GetBulkRequest PDU 格式

新增的 GetBulkRequest PDU 所对应的基本操作类型也是 GetNext 操作，但通过对 Non Repeaters（无后继）和 Max Repetitions（最多后继）参数的设定，可高效率地从 Agent 一次性获取大量被管理对象的参数值，甚至整个对象数据表。主要目的是尽量减少查询大量管理信息所进行的协议交换次数。GetBulkRequest PDU 利用与 GetNextRequest PDU 相同的选择原则，即总是顺序地选择下一个对象，不同的是 GetBulkRequest PDU 可以选择多个后继对象，然后获取多个对象的属性值。

GetBulk Request PDU 的格式如图 11-56 所示。

PDU Type	Request ID	Non Repeaters	Max Repetitions	Variable Bindings

图 11-56 GetBulk PDU 的格式

- Non Repeaters：指出在“变量绑定”字段中有多少个只返回一个后继对象的对象，这些对象一次只能查询一个后继对象，因为这类对象是一个标准变量，即只有一个相关的参数。此时 GetBulk PDU 的应答方式与 GetRequest PDU 一样。
- Max Repetitions：指出在“变量绑定”字段中可以返回多个后继对象的最大值（即一个对象最多可返回的最多后继对象数），以实现一次查询多个后继对象。

下面通过一个示例来体现 GetBulkRequest PDU 与 GetNextRequest PDU 在操作方面的区别。假设有一个变量绑定列表中绑定了一系列的参数变量 A、B、C、D，其中，A、B、D 为标准变量，而 C 下面又分了 C1、C2、C3 三个子变量（假设为一个路由表中的“目的 IP 地址”“下一跳 IP 地址”和“出接口”），且这 3 个子变量下面又各自包含 4 个参数值（假设为 4 条路由表项），即 C1.1、C1.2、C1.3、C1.4，C2.1、C2.2、C2.3、C2.4，C3.1、C3.2、C3.3、C3.4，如图 11-57 所示。

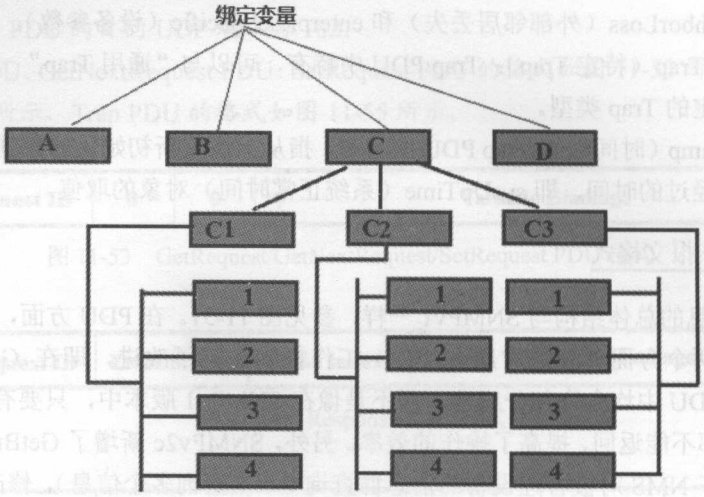


图 11-57 “变量绑定”字段中的变量示例

如果用 GetNextRequest PDU 来操作（用来遍历绑定表中的各变量属性值），则请求和响应操作顺序如下（首先用的是 GetRequest PDU）：

（1）GetRequest(A,B)。

（2）GetResponse (a,b)：假设 A、B 变量的值分别为 a 和 b。

（3）GetNextRequest(C.C1, C.C2, C.C3)。

（4）GetResponse (C.C1.1, C.C2.1, C.C3.1)：返回第一组参数值。

（5）GetNextRequest(C.C1.1, C.C2.1, C.C3.1)：请求获取本组的下一组参数值。

（6）GetResponse (C.C1.2, C.C2.2, C.C3.2)。

（7）GetNextRequest(C.C1.2, C.C2.2, C.C3.2)。

（8）GetResponse (C.C1.3, C.C2.3, C.C3.3)。

（9）GetNextRequest(C.C1.3, C.C2.3, C.C3.3)。

（10）GetResponse (C.C1.4, C.C2.4, C.C3.4)。

（11）GetNextRequest(C.C1.4, C.C2.4, C.C3.4)。

（12）GetResponse (C.C2.1,C.C3.1, D)：C.C1..4 的下一个变量是 C.C.2.1，同理 C.C2.4 的下一个变量是 C.C3.1。

从以上可以看出，如果采用 GetNextRequest PDU 进行操作的话，要频繁交换许多次才能完整地获取各变量的参数值；如果用 GetBulkRequest PDU 来操作的话，就要简单许多，具体如下：

（1）GetBulkRequest(2,3,A,B,C.C1, C.C2, C.C3)：其中 2 代表变量绑定表中有 2 个是标准变量，一次只能返回一个后续对象，3 代表变量绑定表中一个对象最多只能返回 3 个后继对象。

（2）GetResponse (a,b,C.C1.1, C.C2.1,C.C3.1　C.C1.2, C.C2.2,C.C3.2　C.C1.3, C.C2.3,C.C3.3)：因为在请求 PDU 中限制了一个对象最多只能返回 3 个后继对象，所以响应 PDU 中 C1 组只返回

C.C1.1、C.C1.2 和 C.C1.3 这 3 个对象，其他同理。

（3）GetBulkRequest(0,2,C.C1.3, C.C2.3, C.C3.3)：其中的 0 代表本变量绑定表中无标准变量，2 代表一个对象最多返回两个后继对象。

（4）GetResponse (C.C1.4, C.C2.4,C.C3.4　C.C2.1, C.C3.1,D)：因为在前面的请求 PDU 中限制了最多可以返回两个后继对象，所以在 C1、C2 和 C3 组均返回了最后的 C.C1.4、C.C2.4 和 C.C3.4 后还需要继续返回下一个对象，分别为 C.C2.1、C.C3.1、D。

2. Trap PDU 格式

SNMPv2c 修改了 SNMPv1 的 Trap PDU 格式。SNMPv2c Trap PUD 采用 SNMPv1 Get/GetNext/Set PDU 的格式，并将 sysUpTime 和 snmpTrapOID 作为 Variable bindings 中的变量来构造报文。SNMPv2c Trap PDU 的格式如图 11-58 所示。

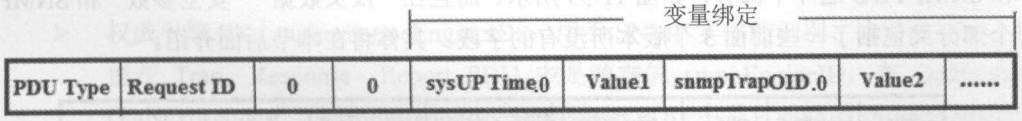


图 11-58　SNMPv2c Trap PDU 的格式

在 SNMPv2 的 Trap PDU 的“变量绑定”列表的头两个变量必须分别是 sysUPTime.0 和 snmpTrapOID.0，分别代表发送 Trap 消息的对象自上次重置到本次 Trap 消息发送所经过的时间和发送 Trap 消息的对象 OID。

11.7.5　SNMPv3 报文格式

SNMPv3 版本是基于 SNMPv2 版本的更新，但与 SNMPv2 版本相比没有定义新的网管操作类型、消息类型和 PDU 结构，主要改进表现在安全方面和消息格式上，但可同时兼容 SNMPv1、SNMPv2 和 SNMPv2c 版本。另外，在 SNMPv3 中管理进程和代理统称为“实体”（Enty），实体是由 SNMP 引擎（Engine）和应用程序（Application）组成的。

1. SNMPv3 PDU

在 PDU 方面，SNMPv3 包括了 8 种 PDU 类型：GetRequest、GetNextRequest、SetRequest、Response、GetBulkRequest、SNMPv2 Trap、InformRequest 和 Report PDU。

【说明】SNMPv1 只有 GetRequest、GetNextRequest、SetRequest、Response 和 trap 这 5 种 PDU，SNMPv2c 在 SNMPv1 版本的基础上增加了 GetBulkRequest PDU，一共有 6 种，SNMPv2 除了没有 Report PDU 外，其余 7 种全有。

InformRequest PDU 也是 Agent 向 NMS 发送的（目的端口为 UDP 162），与 Trap PDU 类似，只是 InformRequest PDU 要求 NMS 必须进行响应。

InformRequest PDU 与 SNMPv2 Trap PDU 结构一样，参见图 11-58，只是“变量绑定”字段中相关的通告消息。收到该请求 PDU 的实体首先检查该 PDU 的消息尺寸，如果超过限度，则用一个含有 toobig 错误代码的 Response PDU 进行响应，否则接收实体将该请求 PDU 的内容转到信息的

目的地(应用实体),同时对发出该请求 PDU 的代理进程用 error-status 字段值为 noError 的 Response PDU 进行应答。

在 RFC 3416 中定义的 Report PDU 是当接收方接收到的消息不能解密时（如由于密钥错误）或者发送方明确指定要向发送端返回请求 PDU“变量绑定”列表中的参数值时发送（此时接收方不是以 Response PDU 进行响应了，而是以 Report PDU 进行响应）。这个 PDU 的格式也与 SNMPv2/v2c 版本的 Trap PDU 一样，但在“变量绑定”中列出的是哪个变量不能解密以及可能的错误原因等，或者原来请求类 PDU 中的各参数值，而且可以设置是否需要在解密出错时向发送方发送相应的 Report PDU。

2. SNMPv3 消息格式

SNMPv3 在消息格式改进方面与图 11-51 有比较大的区别，包括“版本”“报头数据”“安全参数”和 SNMP PDU 这 4 个部分，如图 11-59 所示。而且在“报头数据”“安全参数”和 SNMP PDU 这 3 个部分又包括了一些前面 3 个版本所没有的字段，具体将在本节后面介绍。

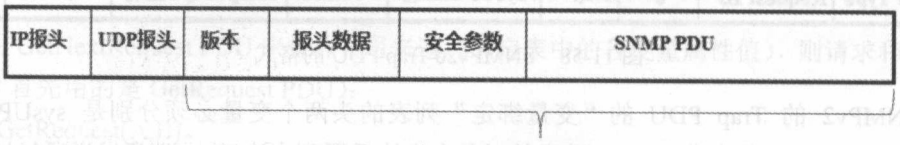


图 11-59　SNMPv3 版本报文格式

- 版本：表示 SNMP 协议的版本号。
- 报头数据：主要包含消息发送者所能支持的最大消息尺寸、消息采用的安全模式等描述内容，格式如图 11-60 所示。

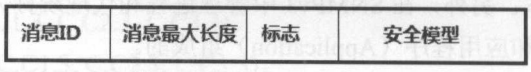


图 11-60　SNMPv3 消息“报头数据”部分的格式

- 消息 ID（MessageID）：消息序列号。
- 消息最大长度（MaxSize）：消息发送者所能容纳的消息最大字节数，同时也表明了发送者能够接收到的最大字节数，类似于 TCP 数据段中的“窗口大小”字段。
- 标志（Flags）：消息标志位，占 1 个字节，只有最低的 3 位（从低到高分别代表认证标志位、加密标志位、Report PDU 标志位）有效。最终的结果有多个，如 0x0（00000000）表示不认证不加密、0x1（00000001）表示认证不加密、0x3（00000011）表示认证加密、0x4（00000100）表示发送 Report PDU 等。当 Report PDU 标志位置 1 时，接收方在收到的消息不能解密（如由于密钥错误）时必须向发送方发送 Report PDU，置 0 时不需要发送；当加密标志位置 1 时启用加密功能，置 0 时不启用；当

认证标志位置 1 时启用认证功能，置 0 时不启用。虽然可以只认证不加密，但不能只加密不认证。

- 安全模型（SecurityModel）：消息的安全模型值，取值为 0～3。0 表示任何模型，1 表示采用 SNMPv1 安全模型，2 表示采用 SNMPv2c 安全模型，3 表示采用 SNMPv3 安全模型。发送方和接收方必须采用相同的安全模型。

- 安全参数：包含 SNMP 实体引擎的相关信息、用户名、认证参数、加密参数等安全信息，格式如图 11-61 所示。

权威引擎ID	权威引擎启动次数	权威引擎时间	用户名	认证参数	加密参数

图 11-61 SNMPv3 消息“安全参数”部分的格式

  - 权威引擎 ID（AuthoritativeEngineID）：用于 SNMP 实体的识别、认证和加密。该取值在 Trap、Response、Report PDU 中是源端的 snmpEngineID，在 GetRequest、GetNextRequest、GetBulkRequest、SetRequest PDU 中是目的端的 snmpEngineID。
  - 权威引擎启动次数（AuthoritativeEngineBoots）：表示从初次配置时开始，SNMP 引擎已经初始化或重新初始化的次数。
  - 权威引擎时间（AuthoritativeEngineTime）：SNMP 正常运行时间，用于时间窗口判断。
  - 用户名（UserName）：进行认证的用户账户名，NMS 和 Agent 配置的用户名必须保持一致。
  - 认证参数（AuthenticationParameters）：认证运算时所需要的密钥。如果没有使用认证则为空。
  - 加密参数（PrivacyParameters）：加密运算时所用到的参数。如果没有使用加密则为空。

- SNMP PDU：SNMPv3 的 PDU 包含三部分内容（统称为 Scoped PDU）。
  - 上下文引擎 ID（Context EgineID）：唯一识别一个 SNMP 实体的标识符。对于流入消息，与 PDU 类型一起决定该 PDU 应该发往哪个应用程序；对于流出消息，该字段值由上层应用提供，代表哪个应用程序。
  - 上下文名（Context Name）：用于唯一识别在相关联的上下文引擎范围内部特定的上下文名称。
  - PDU 数据（PDU data）：包含 PDU 类型、请求标识符、变量绑定列表等信息，包括 GetRequest PDU、GetNextRequest PDU、SetRequest PDU、Response PDU、Trap PDU、GetBulkRequest PDU、InformRequest PDU 和 Report PDU 八种类型。

11.7.6 SNMPv1 和 SNMPv2c 实现机制

SNMPv1 与 SNMPv2c 实现机制基本一致，SNMPv2c 丰富了错误码，新增了 GetBulk 操作。下面以在 SNMPv1 版本环境中执行 Get、GetNext 和 Set 操作为例来描述 SNMPv1 和 SNMPv2c 的实

现机制。

1. Get 操作

假设 NMS 想要获取被管理设备 MIB 节点 sysName 的值（sysName 对象在允许访问视图内），使用 public 为可读团体名（团体名可自行设置），过程如下：

（1）NMS 给 Agent 发送 Get 请求（即发送 GetResquest PDU），请求报文主要字段将被设置为：Version（版本）字段的值为 1（不同厂商设备的版本号定义可能不一样，下同），Community（团体）字段的值为 public，PDU 里“变量绑定”中的 Name1 字段为 sysName.0。

（2）Agent 给 NMS 发送 GetRespone PDU 响应，说明是否获取成功。如果成功，则 Response PDU 里“变量绑定”中的 Value1 字段的值为设备的名字（如 Agent010-H3C）；如果获取失败，则在 Error status 字段填上出错的原因，在 Error index 字段填上出错的位置信息。

2. GetNext 操作

假设 NMS 想要获取被管理设备 MIB 节点 sysName 的下一个节点 sysLocation 的值（sysName 和 sysLocation 对象都在允许访问视图内），使用 public 为可读团体名，过程如下：

（1）NMS 给 Agent 发送 GetNext 请求（即发送 GetNextRequest PDU），请求报文主要字段将被设置为：Version 字段的值为 1，Community 字段的值为 public，PDU 里“变量绑定”中的 Name1 字段为 sysName.0。

（2）Agent 给 NMS 发送 GetNextResponse PDU 响应。如果成功，则 Response PDU 里“变量绑定”中的 Name1 字段值为 sysName.0 的下一个节点 sysLocation.0，Value1 字段的值为地点（如 Beijing China）；如果获取失败，则在 Error status 字段填上出错的原因，在 Error index 字段填上出错的位置信息。

3. Set 操作

假设 NMS 想要设置被管理设备 MIB 节点 sysName 的值为 Device01，使用 private 为可写团体名，过程如下：

（1）NMS 给 Agent 发送 Set 请求（即发送 SetRequest PDU），请求报文主要字段将被设置为：Version 字段的值为 1，Community 字段的值为 private，PDU 里“变量绑定”中的 Name1 字段为 sysName.0，Value1 字段的值填为 Device01。

（2）Agent 给 NMS 发送 SetResponse PDU 响应，说明是否设置成功。如果成功，则 Response PDU 里“变量绑定”中的 Value1 字段的值填为设备的新名字（如 Device01）；如果设置失败，则在 Error status 字段填上出错的原因，在 Error index 字段填上出错的位置信息。

4. Trap 操作

当设备发生某些异常需要通知 NMS 时，Agent 会主动发出 Trap 报文。例如设备某端口网线被拔出，Agent 发送 linkDown 的 Trap 消息给 NMS。Version 字段的值为 1，Community 字段的值为 public，PDU 中 enterprise 字段为 sysObjectID.0 的取值（如为 enterprises.25506），Generic trap 字段值为 linkDown，Variable bindings 字段携带接口相关信息。

11.7.7 SNMPv3 实现机制

SNMPv3 由于采用了 USM（User Security Model）和 VACM（View-based Access Control Model），在安全性上得到了提升：

- USM：在 RFC 3414 中定义，提供身份验证和数据加密服务。实现这个功能要求 NMS 和 Agent 必须共享同一密钥。
  - 身份验证：身份验证是指 Agent 或 NMS 接到信息时首先必须确认信息是否来自有权限的 NMS 或 Agent，并且信息在传输过程中未被改变。RFC2104 中定义了 HMAC 安全哈希函数和密钥来产生信息验证码。

【说明】SNMP 使用的密钥相关的哈希运算消息认证码（Hash-based Message Authentication Code，HMAC）可以分为两种：HMAC-MD5-96 和 HMAC-SHA-96。前者的哈希函数是 MD5，使用 128 位 authKey 作为输入；后者的哈希函数是 SHA-1，使用 160 位 authKey 作为输入。

  - 加密：采用数据加密标准 DES（Data Encryption Standard）的密码组链接 CBC（Cipher Block Chaining）码，使用 128 位的 privKey 作为输入。管理站使用密钥计算验证码，然后将其加入信息中，而代理则使用同一密钥从接收的信息中提取出验证码，从而得到信息。加密的过程与身份验证类似，也需要管理站和代理共享同一密钥来实现信息的加密和解密。
- VACM：在 RFC 3415 中定义，对用户组或者团体名实现基于视图的访问控制。用户必须首先配置一个视图并指明权限。用户可以在配置用户或者用户组或者团体名的时候加载这个视图达到限制读操作或写操作或 Trap 的目的。

SNMPv3 各操作的实现机制同 SNMPv1 和 SNMPv2c 基本一样，主要区别在于 SNMPv3 新增加了认证和加密、解密的处理。下面以 SNMPv3 使用认证和加密方式执行 Get 操作为例来描述其实现机制，过程如下：

（1）NMS 发送不带任何认证和加密参数的 Get 请求（即发送 GetRequest PDU），Flags 字段设置为 0x4（00000100，表示不认证、不加密，但需要向发送方发送 Report PDU），以获取 contextEngineID、contextName、AuthoritativeEngineID、AuthoritativeEngineBoots、AuthoritativeEngineTime 等相关安全参数的值。

（2）Agent 解析消息，然后向 NMS 发送 Report PDU 并携带上述相关参数的值。

（3）NMS 再次给 Agent 发送 Get 请求（即发送 GetRequest PDU）获取 sysName 节点值。此时请求报文中的主要字段将被设置如下：Version 字段的值为 3，将第（2）步获取到的各安全参数值填入相应字段，PDU 里“变量绑定”中的 Name1 字段为 sysName.0（代表想获取 sysName 节点值），且根据配置的认证算法计算出 AuthenticationParameters（认证参数），使用配置的加密算法计算出 PrivacyParameters（加密参数），使用配置的加密算法对 PDU 数据部分进行加密。

（3）Agent 收到 NMS 发来的 GetRequest PDU 后，根据本地的安全参数设置对所接收到的消息进行认证，认证通过后对 PDU 报文进行解密（因为在 GetRequest PDU 中已获取到了安全参数，

所以正常情况下两者是一致的）。解密成功后，获取 sysName.0 对象的值，并将 Response PDU 里“变量绑定”中的 Value1 字段的值填为设备的名字（如 Agent010）。如果认证、解密失败或者获取参数值失败，则在 Error status 字段填上出错的原因，在 Error index 字段填上出错的位置信息。再对 Response PDU 进行加密，设置 contextEngineID、contextName、AuthoritativeEngineID、AuthoritativeEngineBoots、AuthoritativeEngineTime、AuthenticationParameters、PrivacyParameters 等字段参数值，发送 Response PDU 响应报文。

11.8 课后自测题

为了巩固本章所学的内容，请认真做下面的自测题，参考答案见附录。

11.8.1 填空题

1．在应用层中为了解决并发访问，服务器通常采用两种解决方案：一种是按照先进先出的原则对各访问请求进行响应，称为_____；另一种是为每个访问请求创建一个子进程，由子进程分别进行响应，称为_____。

2．Web 服务模型包括 3 个部分，其中提供域名解析服务的是_____，提供 Web 服务的是_____，发起 Web 服务访问的是_____。

3．一个 HTTP 请求报文包括_____、_____、_____和_____四大部分，最前面的部分又包括_____、_____和_____这 3 个字段，它们之间以_____分隔，在最后以一个_____结束。

4．HTTP 响应报文也是由 4 部分组成，分别是_____、_____、_____和_____，最前面的部分也包括_____、_____和_____这 3 个字段，它们之间以_____分隔，在最后以一个_____结束。

5．DHCP 报文主要有 8 类，其中在 DHCP 服务器第一次为 DHCP 客户端分配 IP 地址时可使用的有_____、_____、_____、_____和_____5 种，用来通知 DHCP 服务器释放某个 IP 地址时要使用的是_____，用于通知 DHCP 服务器当前分配的 IP 地址不可用，请求重新分配时所用的是_____，用于从 DHCP 服务器上获取更详细的信息时使用的是_____。

6．DHCP 中继功能是在 DHCP 报文的_____选项中提供的，最多可以包含_____个子选项，但目前设备主要只支持两个子选项，其中用于指示接收到 DHCP 客户端请求报文的 DHCP 中继设备端口所属 VLAN 的 ID 和端口索引的是_____，用于指示接收到 DHCP 客户端请求报文的 DHCP 中继设备的 MAC 地址的是_____。

7．整个电子邮件系统包括两个部分：负责电子邮件用户书写、发送、接收和阅读电子邮件的是_____，位于_____；用于负责电子邮件在发件方和收件方的客户端和服务器端之间数据传输的是_____，位于_____。

8．在我们常用的 SMTP/POP3 电子邮件系统中，“用户代理”部分会同时提供_____和_____服务，而在“消息传输代理”部分，通常会同时提供_____、_____和_____服务。

9．SMTP 协议是采用_____的方式把客户端的邮件发送给邮件服务器，或者把邮件从本端邮件服务器发送给对端邮件服务器，所用的传输层端口是_____，进行邮件发送时包括_____、_____和_____3 个阶段。

10．POP3 协议是邮件客户端采用_____的方式从邮件服务器中接收邮件，所用的传输层端口为_____，进行邮件接收时包括_____、_____和_____3 个阶段。

11．FTP 协议使用的传输层协议是_____，有_____和_____两种工作模式，但不管是哪种工作模式，控制连接建立时 FTP 服务器使用的端口都是_____，_____使用 TCP 20 端口建立数据连接。

12．FTP 的传输模式包括_____、_____和_____3 种，FTP 的数据结构包括_____、_____和_____3 种。

13．要从 Cisco 交换机上通过 FTP 协议把配置文件上传到 FTP 服务器中，最好采用_____或_____数据类型，而要上传 IOS 映象文件到 FTP 服务器上，应选择_____数据类型。

14．SNMP 管理模型包括_____、_____、_____和_____4 个部分，_____是 SNMP 客户端。

15．SNMPv1 包括 5 种 PDU：_____、_____、_____、_____和_____，SNMPv2 新增了_____和_____两种 PDU，SNMPv2c 除了不包括_____外，其他 6 种 SNMPv2 PDU 都包括，SNMPv3 在 SNMPv2 基础上新增了_____。

11.8.2 选择题（可多选）

1．下面关于 URI、URL 和 URN 的关系说法正确的是（　　）。

A．URI 和 URN 是 URL 的扩展　　B．URL 和 URN 是 URI 的扩展

C．URI 和 URL 是 URN 的扩展　　D．以上都不对

2．下面不是 URI 合法格式的是（　　）。

A．mailto:mbox@domain　　B．ftp://user:pass@server/file

C．file://path　　D．http://domain/path

3．下面关于 HTML 说法不正确的是（　　）。

A．HTML 是一种网站网页开发语言

B．HTML 标签必须成对出现

C．在静态 HTML 文档中也可以有像图像、动画这样的动态网页元素

D．动态 HTML 文档必须由服务器程序自动生成，不能由人工手动创建

4．下面关于 HTTP 协议说法不正确的是（　　）。

A．必须工作在 C/S 模式

B．在用户访问网站时必须先与 HTTP 服务器建立 HTTP 连接

C．HTTP 本身不记录用户的每次访问过程，也就是它的“无状态”特性

D．因为 HTTP 是一个无连接的协议，所以通过 HTTP 进行的数据通信是不可靠的

5．在 HTTP 报文中不需要在最后加上 CRLF 的是（　　）。

A．请求行或响应行　　B．请求头部行

C．空行　　D．实体主体行

6．下列 HTTP 请求头中指出了 Web 客户端能处理的 MIME 页面类型的是（　　）。

A．Accept　　B．Accept-Charset

C．Accept-Encoding　　D．Host

7．如果 Web 服务器向客户端返回一个表明请求成功的 HTTP 响应报文，则其状态码为（　　）。

A．100　　B．200

C．300　　D．400

8．如果 Web 服务器向客户端返回一个表明拒绝提供服务的 HTTP 响应报文，则其状态码为（　　）。

A．400　　B．401

C．402　　D．403

9．Web 服务器的 HTTP 响应报文中用来指明 Web 服务器的软件信息的响应头为（　　）。

A．Allow　　B．Server

C．Location　　D．Content-type

10．在 HTTP 请求头中，如果一个请求头中要包含多个值，则这些值之间以（　　）分隔。

A．空格　　B．:（冒号）

C．,（逗号）　　D．/（反斜杠）

11．下面关于 DNS 域名说法正确的是（　　）。

A．DNS 域名不区分大小写　　B．顶级域名是不能由用户申请的

C．二级和三级域名都是可供用户申请的　　D．一级域名就是域根（.）

12．下列关于 DNS 域名结构说法正确的是（　　）

A．DNS 域名必须唯一

B．同一域树中的 DNS 域名必须在同一级别上唯一

C．example.microsoft.com 的域根为.com

D．example.microsoft.com 是一个三级域名

13．下面关于 DNS 名称服务器说法正确的是（　　）。

A．根名称服务器负责的是对顶级名称服务器的解析

B．全球共有 13 台根名称服务器

C．在局域主机 TCP/IP 属性上配置的 ISP DNS 服务器是权威名称服务器

D．在局域主机 TCP/IP 属性上配置的 ISP DNS 服务器是本地名称服务器

14．在 DNS 应答报文中标识资源记录数据类别的字段是（　　）。

A. TYPE　　B. CLASS　　C. RData　　D. QClass

15. 对下列情形下的 DNS 报文传输说法不正确的是（　　）。

A. 大于 512 个字节的 DNS 报文必须使用 TCP 53 号端口传输

B. 从主 DNS 服务器向辅助 DNS 服务器传输的 DNS 报文必须使用 TCP 53 号端口

C. 小于 512 个字节的 DNS 报文必须使用 UDP 53 号端口传输

D. 请求 DNS 报文和响应 DNS 报文必须采用相同的传输层端口协议

16. 下面有关 DNS 的“递归解析”和“迭代解析”说法不正确的是（　　）。

A.“递归解析”是在本地名称服务器解析不了时，后面所有的查询工作由本地名称服务器进行

B.“迭代解析”的所有查询过程均是由 DNS 客户端自己进行

C. 如果客户端的请求报文中没有申请使用递归解析，则默认使用递归解析

D. 如果在本地名称服务器上禁用递归解析，则使用迭代解析

17. 在 DHCP 客户端收到第一个 DHCP 服务器发送的 DHCP Offer 报文后需要立即发送（　　）报文，以通知其他 DHCP 服务器不再向本客户机分配 IP 地址。

A. DHCP REQUEST　　B. DHCP ACK

C. DHCP NAK　　D. DHCP INFORM

18. 下列 DHCP 报文中 Ciaddr 字段为非 0 值的是（　　）。

A. DHCP OFFER　　B. DHCP REQUEST

C. DHCP ACK　　D. DHCP NAK

19. 下列 DHCP 报文中 Siaddr 字段为非 0 值的是（　　）。

A. DHCP OFFER　　B. DHCP REQUEST

C. DHCP ACK　　D. DHCP NAK

20. 下列 DHCP 报文字段中用来指示 DHCP 服务器分配给客户端的 IP 地址的是（　　）。

A. Ciaddr　　B. Yiaddr　　C. Siaddr

D. Giaddr　　E. Chaddr

21. 下列 DHCP 报文字段中用来指示 DHCP 客户端 MAC 地址的是（　　）。

A. Ciaddr　　B. Yiaddr　　C. Siaddr

D. Giaddr　　E. Chaddr

22. 在 DHCP 服务器初次向 DHCP 客户端分配 IP 地址的过程中不需要用到的报文是（　　）。

A. DHCP DISCOVER　　B. DHCP OFFER

C. DHCP REQUEST　　D. DHCP ACK

E. DHCP RELEASE

23. 在 DHCP OFFER 和 DHCP ACK 报文中下列（　　）字段的值不为 0。

A. Ciaddr　　B. Yiaddr

C. Siaddr　　D. Giaddr

24. 下面关于免费 ARP 报文说法不正确的是（　　）。

A. 用来查询目的 IP 地址所对应的 MAC 地址

B. 用来测试是否有其他节点已使用由 DHCP 服务器分配给本地主机的 IP 地址

C. 源 IP 地址和目的 IP 地址都是本机 IP 地址

D. 源 MAC 地址是本机 MAC 地址，目的 MAC 地址是广播 MAC 地址

25. 下面关于 IP 地址续约的说法不正确的是（　　）。

A. DHCP 客户端需要以单播方式向 DHCP 服务器发送 DHCP REQUEST 报文

B. DHCP 客户端需要以单播方式向 DHCP 服务器发送 DHCP DECLINE 报文

C. 如果第一次续约申请失败，则 DHCP 客户端会在租约期限达到 7/8 时再次以广播方式发送 DHCP REQUEST 报文进行续约

D. 如果第一次续约申请失败，则 DHCP 客户端会在租约期限达到 7/8 时再次以单播方式发送 DHCP DECLINE 报文进行续约

E. 如果第二次续约请求还是失败，则 DHCP 客户端原来租约的 IP 地址将自动被释放

26. 在 DHCP 服务器通过 DHCP 中继设备为 DHCP 客户端分配 IP 地址的过程中，采用广播方式发送的报文是（　　）。

A. 客户端发送的 DHCP DISCOVER 报文

B. DHCP 中继设备转发的 DHCP DISCOVER 报文

C. DHCP 服务器发送的 DHCP OFFER 报文

D. DHCP 中继设备转发的 DHCP OFFER 报文

E. DHCP 服务器发送的 DHCP ACK 报文

F. DHCP 中继设备转发的 DHCP ACK 报文

27. 下面关于 FTP 主动模式说法正确的是（　　）。

A. 主动模式下，控制连接和数据连接都是由 FTP 客户端主动发起的

B. 主动模式下，FTP 客户端需要监听一个随机的 TCP 端口

C. 主动模式的数据连接建立时，FTP 客户端使用的端口是 TCP 20

D. 主动模式下，FTP 客户端需要通过 PORT 命令告知 FTP 服务器它所使用的端口号

28. 下面关于 FTP 被动模式说法不正确的是（　　）。

A. 被动模式与主动模式一样，控制连接也是由 FTP 客户端主动发起的

B. 被动模式与主动模式一样，控制连接建立时 FTP 服务器使用的端口为 TCP 21

C. 被动模式的数据连接建立时，FTP 服务器使用的端口为 TCP 20

D. 被动模式下，FTP 客户端不监听任何 TCP 端口

29. 下面关于 FTP 的 PORT 命令说法不正确的是（　　）。

A. PORT 命令只在 FTP 主动模式下使用，后面带有 6 个十进制数字

B. PROT 命令后面的左边 4 个数字分别为源端 IPv4 地址的 4 个字节值

C. 假设 PORT 命令的最后两个数字为 224 和 15，则源 TCP 端口号为 57359

D．PORT 命令是在建立控制连接时由 FTP 客户端发出的

30．如果 FTP 服务器返回一个 230 的响应码，则代表（　　）。

A．命令执行成功　　B．命令执行失败

C．登录成功　　D．服务就绪

31．下面关于 SNMPv3 与以前版本的区别说法正确的是（　　）。

A．以前版本都采用团体名认证方式，SNMPv3 不采用

B．SNMPv3 新增了以前版本所没有的 GetBulkRequest 和 InformRequest 这两种 PDU

C．SNMPv3 新增了 USM 和 VACM，不兼容以前版本

D．SNMPv3 比 SNMPv2 版本支持更多的错误码

32．SNMP 的 4 种操作中，（　　）是由代理发给 NMS 的，且不需要管理者响应。

A．trap　　B．get　　C．get-next　　D．set

33．在通信协议操作方面，SNMP v2 增加了两个新的 PDU，它们是（　　）。

A．GetBulkResponse　　B．GetBulkRequest

C．informResponse　　D．informRequest

34．下列 PDU 由代理发出的是（　　）。

A．GetRequest　　B．SetRequest　　C．GetNextRequest　　D．trap

35．SNMP v2c 的 GetRequest PDU 的语法和语义都与 SNMP v1 的 GetRequest PDU 相同，差别是 SNMP v2c 对应答的处理（　　）。

A．要么所有的值都返回，要么一个也不返回

B．能够部分地对 GetRequest 操作进行应答

C．能够全部地对 GetRequest 操作进行应答

D．可以都不进行应答

36．下面关于 SNMP PDU 发送的接收所使用的 UDP 端口号说法正确的是（　　）。

A．NMS 发送 GetRequest、GetNextRequest、SetRequest 和 GetBulkRequest 时的 UDP 端口为 161

B．代理接收 GetRequest、GetNextRequest、SetRequest 和 GetBulkRequest 时的 UDP 端口为 161

C．代理发送 Trap PDU 时的 UDP 端口为 162

D．NMS 发送 InformRequest PDU 时的 UDP 端口为 162

37．在 SNMPv2c 中可实现以最小的交换次数检索大量的管理信息的 PDU 是（　　）。

A．GetRequest PDU　　B．InformRequest PDU

C．GetBulkRequest PDU　　D．SetRequestPDU

38．在以下几种 PDU 中，不属于 SNMPv1 的是（　　）。

A．GetRequest　　B．GetBulkRequest

C．GetResponse　　D．SetRequest

39. 下面不属于 SNMPv3 PDU“安全参数”部分字段的是（　　）。

A. UserName　　B. AuthoritativeEngineBoots

C. Context EgineID　　D. PrivacyParameters

40. 下面关于 Response 类 PDU 说法正确的是（　　）。

A. 所有 Response 类 PDU 都是代理发送的

B. 所有 Response 类 PDU 发送时都是使用 UDP 161 端口

C. 所有 Response 类 PDU 的接收都是使用 UDP 162 端口

D. InrofmRequest 的 Response PDU 发送时使用的是临时 UDP 端口，接收是 162 端口

附录

各章课后自测题参考答案

第1章

一、填空题

1．二进制　八进制　十进制　十六进制　B　O或Q　D　H　2　8　10　16　八进制

2．$b_{11}b_{10}b_9\ldots\ldots b0$　$b_{11}\times2^{11}+b_{10}\times2^{10}+b_9\times2^9+\ldots\ldots b_0\times2^0$　3　28000H

3．10　1　2　1　0　1　1

4．符号位　真值　正数本身　对除符号位外的其他各位按位取反　正数本身　先对除符号位外的其他各位按位取反，然后在最后加1

5．-22　10010110　11101001　11101010

6．$-127\leqslant X\leqslant127$　$-(1-2^{-7})\leqslant X\leqslant1-2^{-7}$　01001101

7．32　8　127　64　52　1023

8．外码　区位码　机内码　字形码　交换码

二、选择题

题号	1	2	3	4	5	6	7	8	9	10
答案	D	C	C	B	CB	BE	DC	DE	D	BD
题号	11	12	13	14	15	16	17	18	19	20
答案	C	B	B	A	B	A	BD	AB	AG	B
题号	21	22								
答案	D	D								

三、计算题

第 1 题答案：

（1）116D　1864.71875D

（2）574Q　3324.828125Q

（3）31372D　9627.14453125D

（4）1100111001B　10100010101111.1001B

（5）1222Q　22030.5146Q

（6）9CCH　152C.D1EBH

（7）133Q　267.1Q

（8）33BH　E9.AH

（9）1EEH　CB5.88H

（10）122614Q　100522.36Q

第 2 题答案：

（1）101111B　1011101B

（2）1000011B　1011010B

（3）1111110B　100010100B

（4）101B　111B

（5）1001B　10001B

（6）11011B　10111B

（7）11001B　111000B

（8）1101101B　110010B

第 3 题答案：

（1）1101111B　10111011B

（2）10000001B　10101101B

第 4 题答案：

（1）十进制数 20.5 的单精度表示形式为：0 10000011 10010000000000000000000。最后 18 个 0 是为补足 23 位尾数而加的。

（2）X 的存储形式为：0 0111 11011011，Y 的存储形式为：1 1001 01010100。指数是移码形式，尾数为补码形式。

第 2 章

一、填空题

1．计算机主机　资源子网　通信子网　ARPAnet　OSI/RM

2．7　物理层　数据链路层　网络层　传输层　会话层　表示层　应用层

3．硬件系统　软件系统　软件系统　软件系统　硬件系统

4．LAN　MAN　WAN　MAN　LAN　WAN

5．对等网模式　C/S 模式　C/S 模式

6．广播（Brodcast）网络　点对点（P2P）网络　点对多点（P2MP）网络　非广播多路访问（NBMA）网络　广播网络　点对点网络　NBMA 网络

7．星型结构　环型结构　总线型结构　混合型结构　星型结构　网状结构　树状结构

8．IEEE 802.3　IEEE 802.5　IEEE 802.4

二、选择题

题号	1	2	3	4	5	6	7	8	9	10
答案	B	BD	A	D	B	B	D	A	A	AC
题号	11	12	13	14	15	16	17	18	19	
答案	D	AD	B	C	D	BD	C	D	A	

第3章

一、填空题

1．ARPAnet　IBM　SNA

2．传输层　会话层　表示层　应用层　物理层　应用层　表示层

3．网络接口层　网际层　传输层　应用层　会话层　表示层　应用层

4．MAC　LLC　MAC　LLC

5．SAP　语义　语法　同步

二、选择题

题号	1	2	3	4	5	6	7	8	9	10
答案	C	DE	AB	C	AB	C	A	BC	A	B
题号	11	12	13	14	15	16	17	18	19	20
答案	BCD	B	A	C	D	C	D	C	C	BD
题号	21	22	23	24	25					
答案	A	B	AD	B	B					

第4章

一、填空题

1．源站点　传输系统　目的站点

2. 串行传输　并行传输　数据块　面向字符　面向比特　面向字节　字符
3. 有限带宽、无噪声信道的最大数据传输速率与信道带宽的关系　2　MaxRB=2W
4. 在有随机热噪声的信道上传输数据信号时，最大数据传输速率与信道带宽、信噪比的关系为 $R_{max}=B\times\log_2(1+S/N)$
5. 信号的采样　信号的重建　2　5～10
6. 电平值大于或等于振幅的 1/2　电平值小于振幅的 1/2　电平为正的信号　电平为负的信号
7. AMI 码　CMI 码　曼彻斯特码　差分曼彻斯特码
8. 一个信号码用两个电平来表示　CMI 码　曼彻斯特码　差分曼彻斯特码
9. 模拟法　键控法　非相干解调法（或包络检波法）　相干解调法（或同步检测法）　直接调频法　移频键控法　鉴频法、过零检测法　非相干解调法（或滤波检测法）　相干解调法（或差分检波法）
10. APSK　DPSK　APSK　模拟调相法　键控调相法　相干解调法　极性比较法（相干解调）　相位比较法（差分解调）

二、选择题

题号	1	2	3	4	5	6	7	8	9	10
答案	B	D	C	B	AD	BD	A	B	C	AC
题号	11	12	13	14	15	16	17	18	19	20
答案	BDE	BDE	B	C	DE	D	C	A	C	B
题号	21	22	23	24	25					
答案	A	E	C	A	BCD					

第5章

一、填空题

1. 机械特性　电气特性　功能特性　规程特性　非平衡型　新的非平衡型　平衡型
2. 导向性传输介质　非导向性传输介质　导向性传输介质　非导向性传输介质
3. 100 米　180 米　500 米　2 公里　50Ω
4. 100MHz　100Mb/s　250MHz　1000Mb/s　500MHz　10Gb/s
5. 阶跃型光纤　渐变型光纤　单模环形光纤　单模 W 形光纤　单模三角形光纤　单模椭圆形光纤　单模光纤　多模光纤
6. XFP　WDM
7. 2.4GHz　11Mb/s　11　1、6 和 11 号信道　2 和 7 号信道　3 和 8 号信道　4 和 9 号信道　5 和 10 号信道

8．5GHz　54Mb/s　24　12

9．2.4GHz　5GHz　54Mb/s　13　1、6 和 11 或 13 号信道　2、7 和 12 号信道
3、8 和 13 号信道　4 和 9 号信道　5 和 10 号信道

10．5GHz　3.2Gb/s　7Gb/s　60GHz　7Gb/s

11．4×4 MIMO　8×8 MIMO　10×10 MIMO

二、选择题

题号	1	2	3	4	5	6	7	8	9	10
答案	A	C	AD	D	CD	C	AB	CD	BC	C
题号	11	12	13	14	15	16	17	18		
答案	B	CD	C	BCD	B	D	B	D		

第 6 章

一、填空题

1．MAC　逻辑链路的建立和释放　提供与网络层交互的接口　数据传输差错控制　LLC

2．数据链路管理、封装成帧、透明传输、差错控制

3．有确认的面向连接服务、有确认的无连接服务、无确认的无连接服务
有确认的面向连接服务　数据链路建立、数据传输、数据链路释放　有确认的无连接服务
无确认的无连接服务

4．字节计数法　字符填充的首尾定界符法　比特填充的首尾定界符法　违法编码法

5．SYN　ETX　01111110

6．差错检测　差错纠正　PCC　CRC　反馈检测　计时器　自动重发请求　纠错码

7．接收方必须把所接收到的数据帧完整地返回给发送方　计时器　帧序列号

8．空闲重发请求　连续重发请求　发送方每发送一帧都要停下来等待接收方的确认返回
信道有效利用率低　连续发送一系列信息帧　回退 N 帧　选择重发

9．计算校验位数　确定校验码位置　确定校验码　实现校验和纠错　2^n　1　0

10．XON/XOFF　滑动窗口机制　XON/XOFF

11．正常响应方式（NRM）、异步响应方式（ARM）　异步平衡方式（ABM）
扩充正常响应方式（SNRM）　扩充异步响应方式（SARM）　扩充异步平衡方式（SABM）

12．标志字段 01111110　不断地发送标志字段　0

13．00000011　0x7D　(0x7D,0x5E)　01111101 01011110

二、选择题

题号	1	2	3	4	5	6	7	8	9	10
答案	A	B	D	C	C	A	B	C	BC	A
题号	11	12	13	14	15	16	17	18	19	20
答案	C	A	B	AB	D	AB	C	ABD	ACE	C
题号	21	22	23	24	25	26	27	28	29	30
答案	B	D	B	D	B	B	A	D	D	BD
题号	31	32	33	34	35					
答案	BC	C	D	A	B					

三、计算题

1. 奇校验：111011001010　偶校验：111011001011
2. 10010111001010
3. 奇校验：111110010101　偶校验：001010000101

第7章

一、填空题

1. 多播帧（或组播帧）　单播帧　广播帧
2. CSMA　CSMA/CD　CSMA/CA
3. 1-坚持 CSMA　非-坚持 CSMA　p-坚持 CSMA
4. 冲突检测　冲突阻止　先听后说　一旦冲突，立即停说　等待时机，然后再说　侦听发送数据
5. 截断二进制指数退避算法　51.2μs　15　10　10
6. DSAP　SSAP　Crontrol　0xAA　0x03
7. 类型　x600（或十进制 1536）　长度　x5DC（或十进制 1500）
8. MDI 子层　PMD 子层　PMA 子层　PCS 子层　MII 接口　RS 子层
9. 1000Base-LX　1000Base-SX　1000Base-LH　1000Base-ZX　1000Base-LX10　1000Base-BX10　1000Base-CX　1000Base-T　1000Base-TX
10. MII　GMII　8B/10B　PAM-5　4B/5B
11. 10GBase-SR　10GBase-LR　10GBase-LRM　10GBase-ER　10GBase-ZR　10GBase-LX4　10GBase-CX4　10GBase-KX4　10GBase-KR　10GBase-T　10GBase-SW　10GBase-LW　10GBase-EW　10GBase-ZW　10GBase-X　10GBase-R　10GBase-W
12. 自治式架构　集中式架构

13. RTS CTS Beacon

二、选择题

题号	1	2	3	4	5	6	7	8	9	10
答案	C	B	C	E	C	B	AB	B	D	AB
题号	11	12	13	14	15	16	17	18	19	20
答案	AB	D	A	C	A	D	C	A	C	D
题号	21	22	23	24	25	26	27	28	29	30
答案	CD	AB	B	DEF	AC	B	D	D	A	ACD
题号	31	32	33	34	35	36	37	38	39	40
答案	E	B	DEF	B	D	D	BC	C	A	BD

第8章

一、填空题

1. 线路交换 存储－转发 线路建立 数据传输 线路释放 报文交换 分组交换
2. 虚电路建立 数据传输 虚电路释放 数据分组 呼叫分组
3. 20 60 65535 32 位（或 4 个字节）
4. 基本报头 扩展报头 40 头部长度 标识 标志 分段偏移 校验和 选项 填充 流标签
5. 逐跳选项头 目的选项头 路由头 分段头 认证头 封装安全有效载荷头 上层头
6. 0x00000000 0x0800 0xFFFFFFFF 0x0806
7. MAC 地址 IP 地址 IP 地址 MAC 地址 静态 ARP 表项 动态 ARP 表项 长静态 ARP 表项 短静态 ARP 表项 设备所属 VLAN 的 VLAN ID 所连设备端口
8. 静态路由 扩散法 随机走动法 最短路径法 基于流量的路由法 距离矢量路由算法 链路状态路由算法 分级路由算法
9. 网络 ID 子网掩码 下一跳地址/接口
10. 每个分组头部中携带一个跳数计数器 为所发送的每个数据分组分配一个不同的序号
11. 缓冲区预分配法 数据包丢弃法 定额控制法 存储转发死锁 重装死锁 每个结点设置 m+1 个缓冲区，并顺序编号 使每个数据包上都携带一个全局性的唯一的“时间戳” 没有空闲缓冲区而造成既无法接收也无法转发数据包
12. 转发表 路由表 ARP 表

二、选择题

题号	1	2	3	4	5	6	7	8	9	10
答案	D	AD	D	E	B	A	B	B	A	A
题号	11	12	13	14	15	16	17	18	19	20
答案	C	D	C	B	D	C	ABC	DE	DE	AB
题号	21	22	23	24	25	26	27	28	29	
答案	D	D	B	A	BC	B	C	A	C	

第9章

一、填空题

1. 0　10　110　1110　11110

2. 网络广播地址　子网广播地址　全子网定向广播地址　有限广播地址　255.255.255.255

3. 10.0.0/8　172.16.0.0/12　192.168.0.0/16

4. 单播地址　多播地址　任播地址　链路本地地址　唯一本地单播地址　全球单播地址　特殊地址　兼容性地址

5. 未指定地址　环回地址　::/128　::1/128　IPv4 兼容地址　IPv4 映射地址　6to4 地址

二、选择题

题号	1	2	3	4	5	6	7	8	9	10
答案	D	AB	C	B	D	A	C	BD	B	AC
题号	11	12	13	14	15	16	17	18	19	20
答案	A	B	BC	DE	B	B	D	D	D	A
题号	21	22	23	24	25	26	27	28	29	30
答案	CD	A	A	C	D	B	D	B	AB	AD
题号	31	32	33	34	35	36	37	38	39	40
答案	A	D	B	C	ABC	C	CD	A	D	B
题号	41	42	43	44	45	46	47	48	49	50
答案	B	AB	C	AD	C	D	BC	D	AC	A
题号	51	52	53	54	55	56	57	58	59	60
答案	A	C	B	C	B	B	BD	D	AB	D

第10章

一、填空题

1. 面向连接 TCP 无连接 UDP
2. 传输连接服务 数据传输服务 A C B
3. CR CC DT AK ED EA DR DC
4. 监听（Listen） 连接（Connect） 发送（Send） 接收（Receive） 断开连接（Disconnect） 请求（Request） 指示（Indication） 响应（Response） 确认（Confirm）
5. T-CONNECT.request T-CONNECT.indication T-CONNECT.response T-CONNECT.confirm T-DATA.request T-DATA.indication T-EXPEDITED-DATA.request T-EXPEDITED-DATA.indication T-DISCONNECT.indication T-DISCONNECT.request
6. 进程服务器方案 进程服务器 名称服务器方案 名称服务器
7. 非重复 TSAP 超时连接表 分组的 TTL 机制 三次握手机制
8. SOCKET BIND LISTEN ACCEPT SOCKET CONNECT SEND RECV
9. 字节编号机制 数据段确认机制 超时重传机制 选择性确认
10. 慢启动（Slow start） 拥塞避免（Congestion avoidance） 快速重传（Fast retransmit） 快速恢复（Fast Recovery）

二、选择题

题号	1	2	3	4	5	6	7	8	9	10
答案	B	E	CDE	E	A	A	BDE	BD	ABD	CE
题号	11	12	13	14	15	16	17	18	19	20
答案	DF	D	BE	D	C	B	CD	D	A	C
题号	21	22	23	24	25	26	27	28	29	30
答案	B	AF	C	C	A	A	D	B	A	D
题号	31	32	33	34	35	36	37	38	39	40
答案	D	D	D	A	AB	A	C	B	D	B

第11章

一、填空题

1. 迭代服务器方案 并发服务器方案
2. Web 服务注册中心 Web 服务提供者 Web 服务请求者
3. 请求行（Request Line） 请求头部（Request Header）行 空行

实体主体（Entity Body）行　请求方法　URL　协议版本　空格　CRLF

4. 响应行（Response）　响应报头（Response Header）行　空行
   实体主体（Entity Body）行　协议版本　状态码　描述短语　空格　CRLF
5. DHCP DISCOVER　DHCP OFFER　DHCP REQUEST　DHCP ACK　DHCP NAK
   DHCP Release　DHCP DECLINE　DHCP INFORM
6. Option 82　255　sub-option 1（Circuit ID，电路 ID 子选项）
   sub-option 2（Remote ID，远程 ID 子选项）
7. 用户代理（User Agent）　消息传输代理（Message Transfer Agent）　邮件客户机
   邮件服务器
8. SMTP 客户端　POP3 客户端　SMTP 服务器　POP3 服务器　SMTP 客户端
9. 推　TCP 25　连接建立　邮件传输　连接释放
10. 拉　TCP 110　连接建立　邮件接收　连接释放
11. TCP　主动　被动　21　主动模式
12. 流模式　块模式　压缩模式　文件结构　记录结构　页结构
13. ASCII　EBCDIC　IMAGE
14. NMS　代理　管理对象　MIB　NMS
15. GetRequest　GetNextRequest　SetRequest　Response　Trap　GetBulkRequest
    InformRequest　InformRequest　Report

二、选择题

题号	1	2	3	4	5	6	7	8	9	10
答案	B	C	AB	BD	D	A	B	D	B	C
题号	11	12	13	14	15	16	17	18	19	20
答案	AB	BD	AD	B	C	C	A	C	AC	B
题号	21	22	23	24	25	26	27	28	29	30
答案	E	E	B	A	BD	ADF	D	C	D	C
题号	31	32	33	34	35	36	37	38	39	40
答案	A	A	BD	D	B	BD	C	B	C	D

Computer Network
Technologies in Depth

深入理解
计算机网络

销售分类: 网络与通信/计算机网络

微信号：Waterpub-Pro
唯一官方微信服务平台

ISBN 978-7-5170-4506-9
9 787517 045069
定价:98.00元